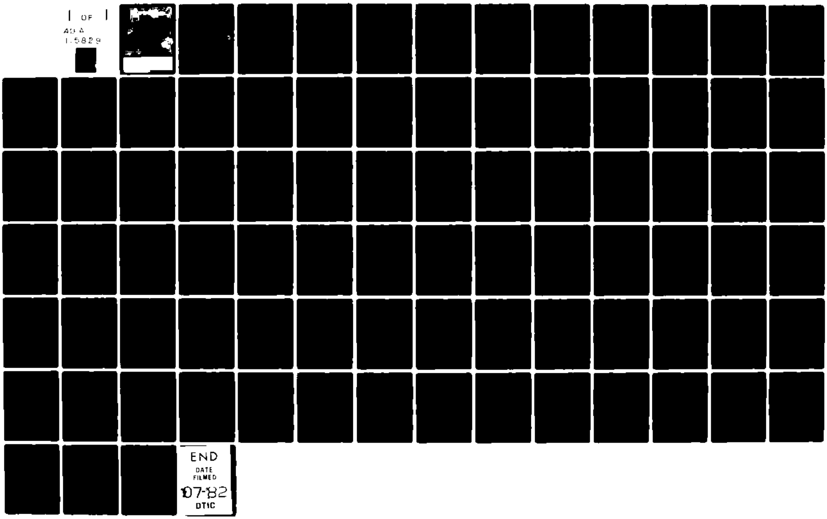
AD-A115 829 CACI INC-FEDERAL ARLINGTON VA F/G 9/2
PROGRAMMER'S MANUAL FOR THE FORECAST 90 COMPUTER PROGRAMS.(U)
1975 R E HAYES DAAG39-75-C-0106

UNCLASSIFIED NL

1 OF 1
AD A
115829



END
DATE
FILMED
07-82
DTIC

1

FINAL TECHNICAL REPORT

PROGRAMMER'S MANUAL FOR THE FORECAST 90 COMPUTER PROGRAMS

Sponsored by:
Department of the Army
Harry Diamond Laboratories

Contract Number	DAAG39-75-C-0106
Program Code Number	W74G3D
Contractor	CACI, Inc.-Federal 1815 North Fort Myer Drive Arlington, Virginia 22209
Effective Date of Contract	November 7, 1974
Expiration Date of Contract	November 7, 1975
Amount of Contract	$224,205
Principal Investigator	Dr. Richard E. Hayes

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Army, Defense Advanced Research Projects Agency, or the U.S. Government.

Programs [illegible] include

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

PREFACE

Three manuals have been written for the FORECAST 90 Project, a contract jointly funded by the Army and the Defense Advanced Research Projects Agency.

- "A Guide to Network Construction and Utilization"
- "User's Manual for the FORECAST 90 Computer Programs"
- "Programmer's Manual for the FORECAST 90 Computer Programs"

These manuals show how to construct and use networks, how to use the computer programs written for the project, and how to maintain the FORECAST 90 computer programs.

A large number of individuals have contributed significantly to the FORECAST 90 Project. Special mention must be given to Colonel John G. Pappageorge, the project monitor at the Strategic Studies Institute, who formulated the initial concept of FORECAST 90 and followed it through the contract phase with uncommon dedication, insight, and patience. The entire research effort is appreciably better as a result of his many comments, criticisms, and suggestions. Colonel Joseph Pizzi, the Director of the Strategic Studies Institute and Chairman of the Study Advisory Group (SAG), provided assistance and guidance at critical points in the project. Members of and observers to the SAG participated heavily in the research, often raising fundamental questions about the project, and always contributing to a better product. Captain Daryl Steiner and Lieutenant Ron Parker of the ADP Support Group at Carlisle Barracks spent many long hours unraveling the undocumented intricacies of the U.S. Army War College computer system.

CACI's support staff edited and typed draft after draft of the three manuals with unfailing good humor. Particular thanks are due to Carol Franco, who converted dangling participles and split infinitives into more readable prose, and Sharon O'Rourke, who always found some new way to juggle work loads so that one more part of the three manuals could be completed. Ann Yamat cheerfully typed most of the drafts, with considerable assistance at critical points from Nancy Streeter. We owe a substantial debt of gratitude to each of these individuals.

INTRODUCTION

CACI, Inc., has written and placed 52 networks in computer storage at the U.S. Army War College (USAWC) as part of the Strategic Studies Institute's FORECAST 90 effort. This manual presents a brief overview of the structure and coding of the 52 FORECAST 90 networks, (which are described in greater detail in "A Guide to Network Construction and Utilization"), and provides documentation for the FORECAST 90 computer programs.

WHAT ARE THE FORECAST 90 NETWORKS?

The 52 FORECAST 90 networks are a way of looking at the subsequents, or downstream effects, of an action that might occur today. Thus, they provide a means to assess the implications of various policy options that might be taken in response to an event that is significant enough to affect existing relationships between countries. These disruptive events are called "catalytic events" in the FORECAST 90 system, and each FORECAST 90 network is built on the occurrence of a specific catalytic event.

Figure 1 presents an example of the structure of the FORECAST 90 networks. Each network fits this form because each is built on a trend, a determining factor, a catalytic event, five affected activities, one or more first-, second-, and third-order effects, and a series of decisional outcomes. The nets are developed to examine a catalytic event on a major ongoing trend in world affairs (e.g., detente) for five affected activities--U.S./USSR relations, U.S./PRC relations, U.S./Japanese relations, U.S./Western European relations, and U.S./other country relations. The impact of the catalytic event on the trend in each of these five affected activities is filtered through one or more first-, second-, and third-order effects (that is, attempts tc develop plausible responses that the major countries involved in the catalytic event or affected by it might attempt to take). The impact of the entire sequence displayed in Figure 1 is

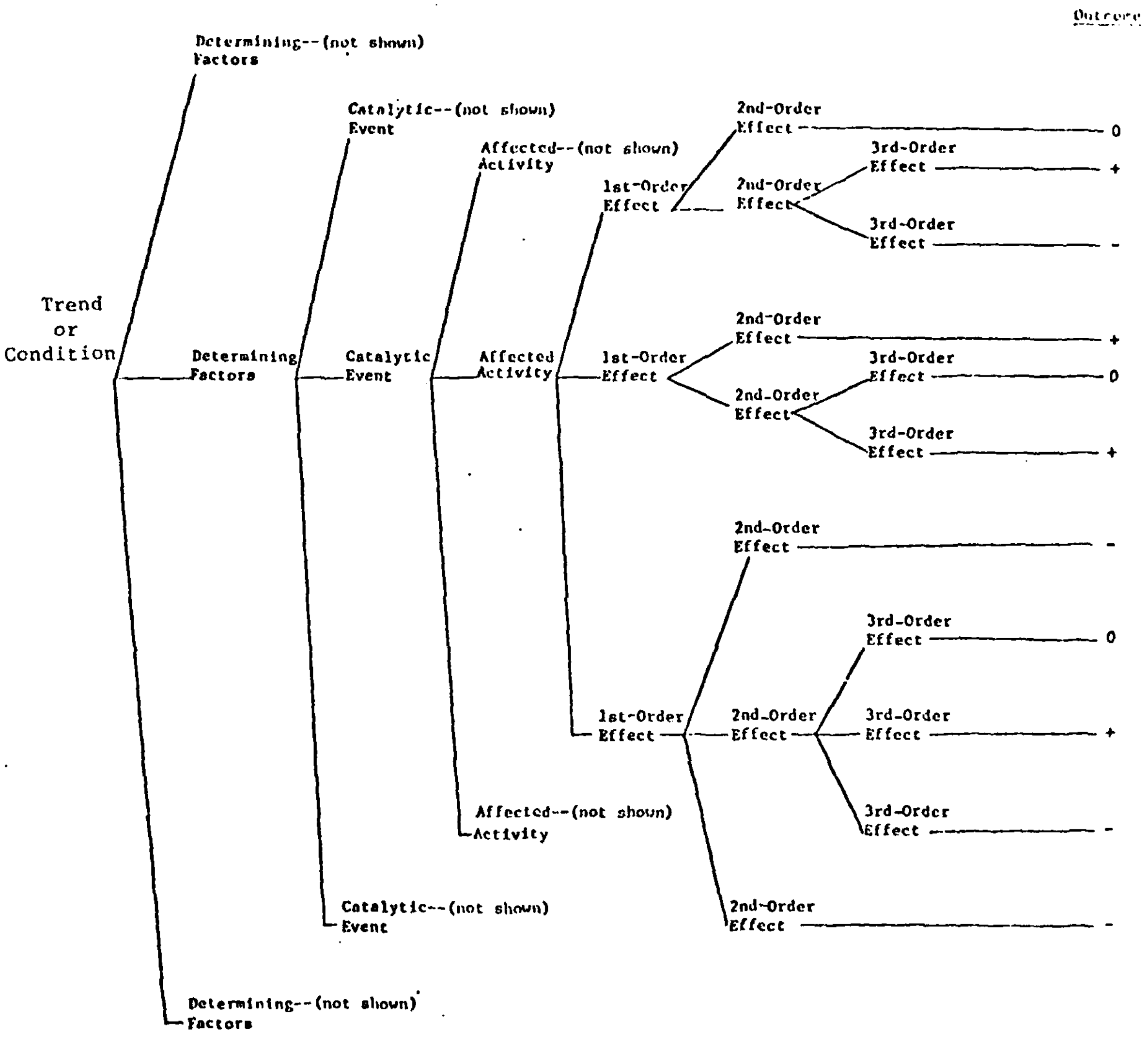


Figure 1. The Structure of the FORECAST 90 Networks.

then summarized in the decisional outcome column where the trend under examination can be increased, decreased, or maintained.

Five different trend areas--economic, military, political, socio-economic and technological--have been used to construct the networks. Regardless of the trend involved, all networks have the same structure.

CODING THE NETWORKS

Each distinct point in a network is called a node, and each node is coded with a unique alphabetic and numeric combination. While the structure of the networks and the structure of the codes are consistent across all of the networks, each node in each network is uniquely numbered. Figure 2 attempts to illustrate these points by displaying the structure of the code for the nodes for a hypothetical political tree.

The code for each network begins with a letter designating whether it is an economic (E), military (M), political (P), socio-pyschological (S), or technological (T) tree. Once the subject area of the network is designated, two digits are added to the code for the determining factor. Two additional digits are added for the catalytic event. One digit each is added for the affected activities, the first-, second-, and third-order effects. The network code ends with a unique one-letter and three-digit designation for each decisional outcome. Using the information given in Figure 2, the node numbers for each point in the network can be written as in Table 1.

Three characteristics of the coding system should be noted. First, each network is stored in the USAWC computer system under the number of its catalytic event. Hence, to reference a computer-stored network, the user must know the number of its catalytic event. For easy reference, the network number and the catalytic event for each network are listed, by category of subject matter, in Appendix I of the "User's Manual for the

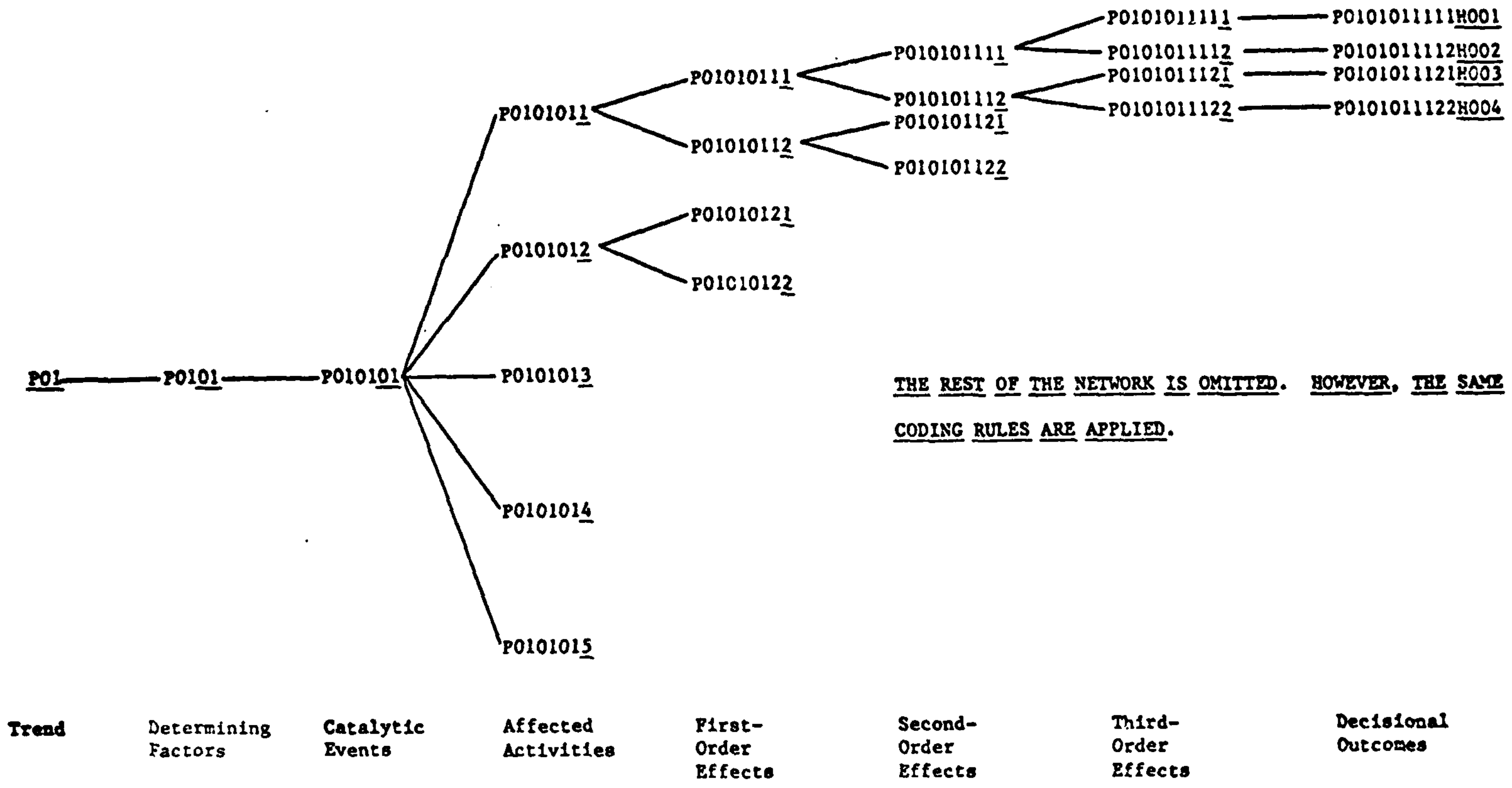


Figure 2: An Example of the Network Coding System from Trend to Decisional Outcomes.

TABLE 1

Sample Code Designations

Level of the Tree	Full Node Designation
Trend	P01.
Determining Factors	P0101.
Catalytic Event	P010101.
Affected Activities	P0101011.
First-order Effects	P01010111.
	P01010112.
Second-order Effects	P010101111.
	P010101112.
Third-order Effects	P0101011111.
	P0101011112.
Decisional Outcome	P0101011111.H001
	P0101011111.H002

FORECAST 90 Computer Programs."[1] Thus, should any potential user of the FORECAST 90 nets ever need to know the number of a specific network, Appendix I of that manual gives the information.

Second, each network employs a standard referencing procedure for the five affected activities on which the networks are focused. These codes, as noted in Figure 2, are standardized as follows:

- U.S./USSR relations = 1
- U.S./PRC relations = 2
- U.S./Japanese relations = 3
- U.S./Western Europe relations = 4
- U.S./other country relations = 5

[1] See Chapter 3 of "A Guide to Network Construction and Utilization" for details on the assignment of these numbers.

Regardless of the network that is being used, the code for the section of the tree on relations between the United States and the Soviet Union will always end in 1.

Third, the node numbers are unique to each node and must be treated as such. Hence, if any digit in the node number is transposed or omitted, the user will not obtain the node that is being sought. Moreover, if the period--as demonstrated in Table 1--is omitted, the user will not receive the node that is being sought. It is extremely important to enter the node exactly since any variation will create errors.

EQUIVALENCE CODING

In addition to a code for each node in the networks, a second code was developed for each of the first-, second-, and third-order effects in each network. A similar code was developed for each catalytic event that has been networked. The purpose of these codes is to help find cross-over points to permit the user to cross-over from one network to another when occurrences in one of the networks have implications for occurrences in a second network.

In developing the equivalence code, seven pieces of information were coded for the contents of the node to attempt to summarize the occurrences there.

- The first actor (the primary initiator of the actions described in the node).
- The second actor (the secondary initiator of the actions described in the node).
- The action taken in the node (activities described in this specific node).
- The first target (the country toward which the action in the node is primarily directed).
- The second target (the country toward which the action in the node is secondarily directed).

- Geographic region (where the action described in the node occurred).
- Substantive topic (subject about which the exchange described in the node occurred).

This information has been coded for each first-, second-, and third-order effect in each network. The codes used can be found in Appendices III through VII of the "User's Manual for the FORECAST 90 Computer Programs."[2]

Once the nodes were coded for these seven pieces of information, equivalence was sought to find nodes that had the same actor, action, target, geographic region, and substantive topic. Once commonly structured nodes were found, they formed cross-over points that were used to link together different networks. Figure 3 displays two unrelated networks that have commonly structured nodes (designated by letters). Figure 4 shows the use of the cross-over points to join--or "integrate"--the two separate networks.

All of the 52 networks currently available for FORECAST 90 have been examined for cross-over points and integrated. Where a node in one network crosses to another node in a second network, a particular statement--called a GO TO statement--is used to designate the occurrence. Thus, when the user prints a part of a network on the computer he may see GO TO statements at one or more nodes that are followed by a number. The number given refers to the location in the same tree or in another tree to which the first node is to branch. Additionally, at the end of each printing of a network section, the user can obtain a list of the GO TO statements encountered in that printing. This list of statements shows where the cross-overs have occurred and indicates what trees should be examined to print the nodes to which the cross-overs have been made.

[2] A more detailed discussion of network integration is found in Chapter 4 of "A Guide to Network Construction and Utilization."

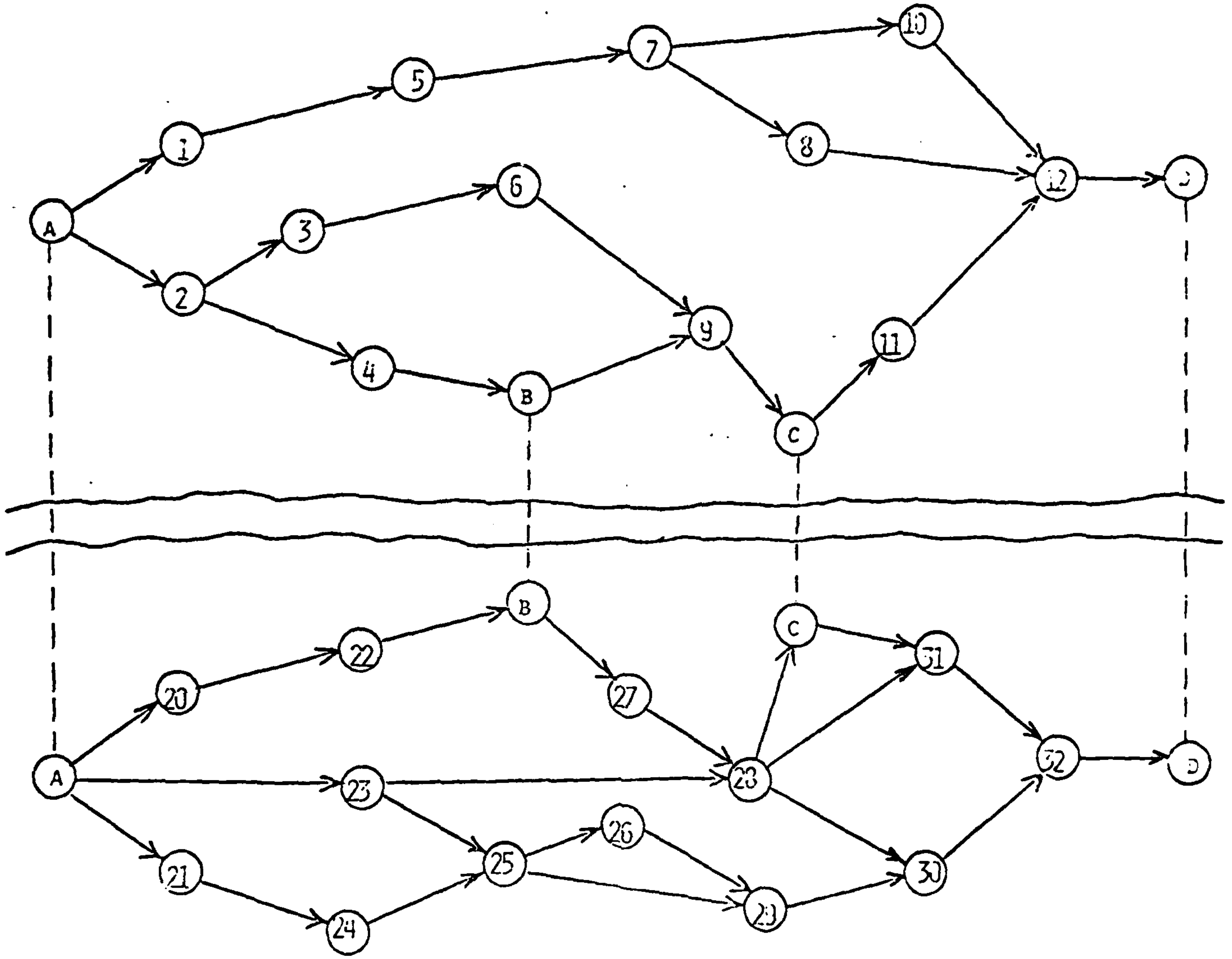


Figure 3. Two Distinct Networks with Potential Cross-Over Points

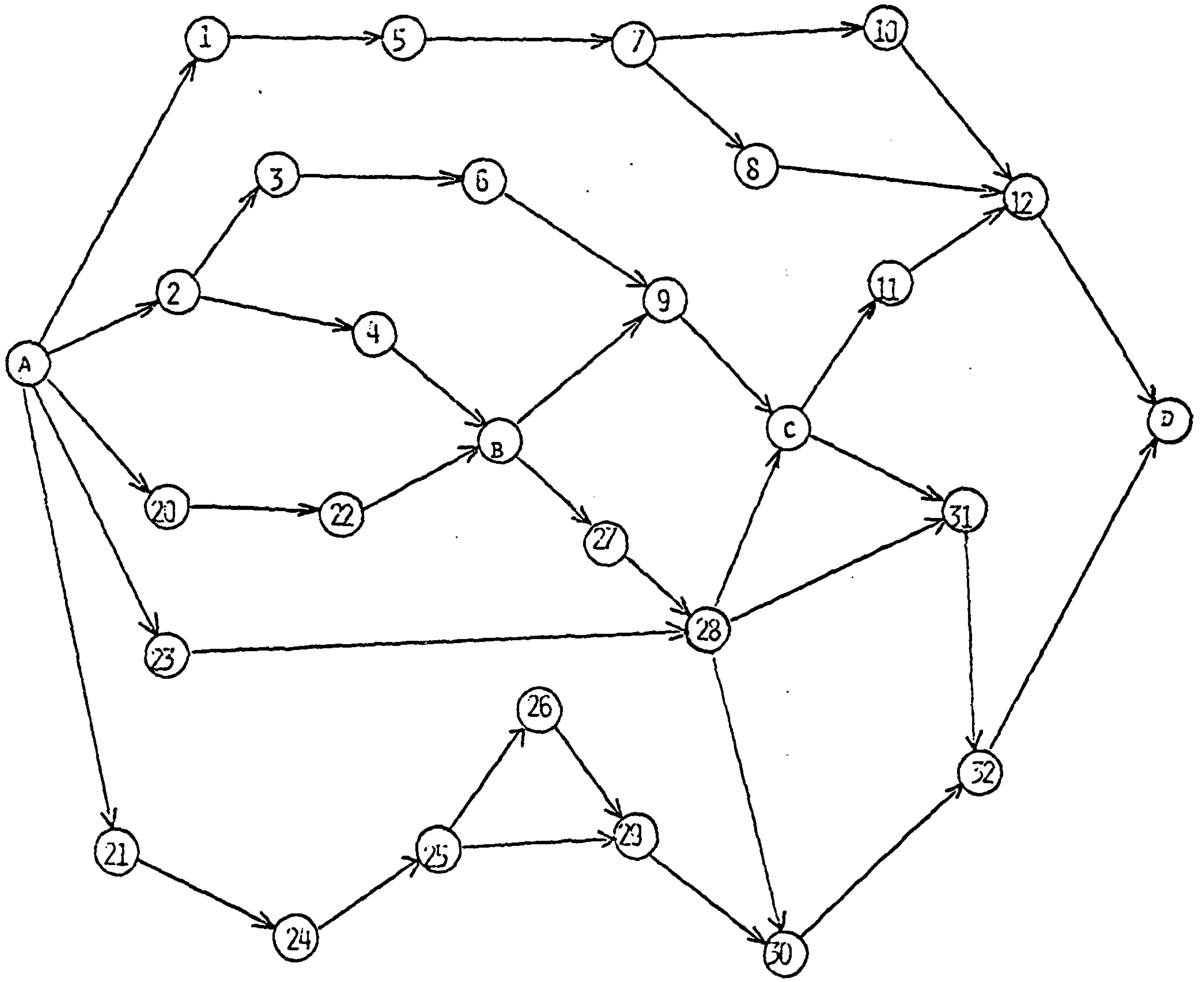


Figure 4. Two Integrated Networks

GENERAL INFORMATION ON THE FORECAST 90 PROGRAMS

Four computer programs have been developed for use with the FORECAST 90 networks and related data files. These programs are written in FORTRAN and, except for NET2, are interactive. Use of the programs is described in the companion "User's Manual for the FORECAST 90 Computer Programs." The program documentation given in this manual will include a general statement on the programming philosophy, a discussion of the structure of the required data file, and description of each program (including operational-level flow diagrams, definitions of the variables and annotated program listings in the Appendices).

Table 2 presents a brief summary of information on the FORECAST 90 programs.

TABLE 2

Summary of FORECAST 90 Programs

Program Name	Type	Objectives	Input File	Function
NET1	Interactive	Search a data file for the structure of a single network.	Network Files. Each file contains node designations and text for all nodes in a single network. There are 52 files with between 54-423 nodes per network.	To print portions of a single network selected by the user.
NET2	Batch	Search a data file for all nodes with an equivalent structure.	Node Information File. Each file record provides summary information on the structure of action in each node in each network. There are presently 5000 records in the file.	To identify and print for each node of each network all nodes in that network and in other networks that have the same action structure.
NET3	Interactive	Search a data file for all nodes with a structure that is equivalent to the one specified by the user.	Node Information File.	To identify and print information on the location of equivalence structure specified by the user as found in any of the 52 networks.
NET4	Interactive	Summarize the status of all networks presently stored in the system.	Tree Summary File. Information is included for each network on the substance of the net, the date of the net, the date of completion, last date of revision, and number of nodes in the network.	To print information on the content and status of the networks selected through criteria specified by the user.

GENERAL PROGRAMMING PHILOSOPHY

Figure 5, which shows a very simple tree, indicates the way in which the terms "node" and "level" will be used. It is seen there that a node is a branch point in a "tree." Each tree has 8 levels. The first three levels have one node each; level 4 has five nodes; the number of nodes in the higher levels may vary from tree to tree, but the number of nodes in levels 7 and 8 is always the same within a single tree.

Each node is identified by a 7-character node designation. The first character of the node designation is always one of the letters: E (economic), P (political), M (military), S (socio-psychological), or T (technological). Examples are shown below:

Level	Full Node Designation	Unique Part of the Node Designation
1	P01.	-[a]
2	P0101.	-
3	P010101.	-
4	P0101011.	1
5	P01010111.	11
	P01010112.	12
6	P010101111.	111
	P010101121.	121
7	P0101011111.	1111
	P0101011211.	1211
8	P0101011111.H001	1111(1)[b]
	P0101011211.H002	1211(1)

[a] Since there is only one node in each of the first three levels, no unique designation is necessary.

[b] The last 1 is added by the program.

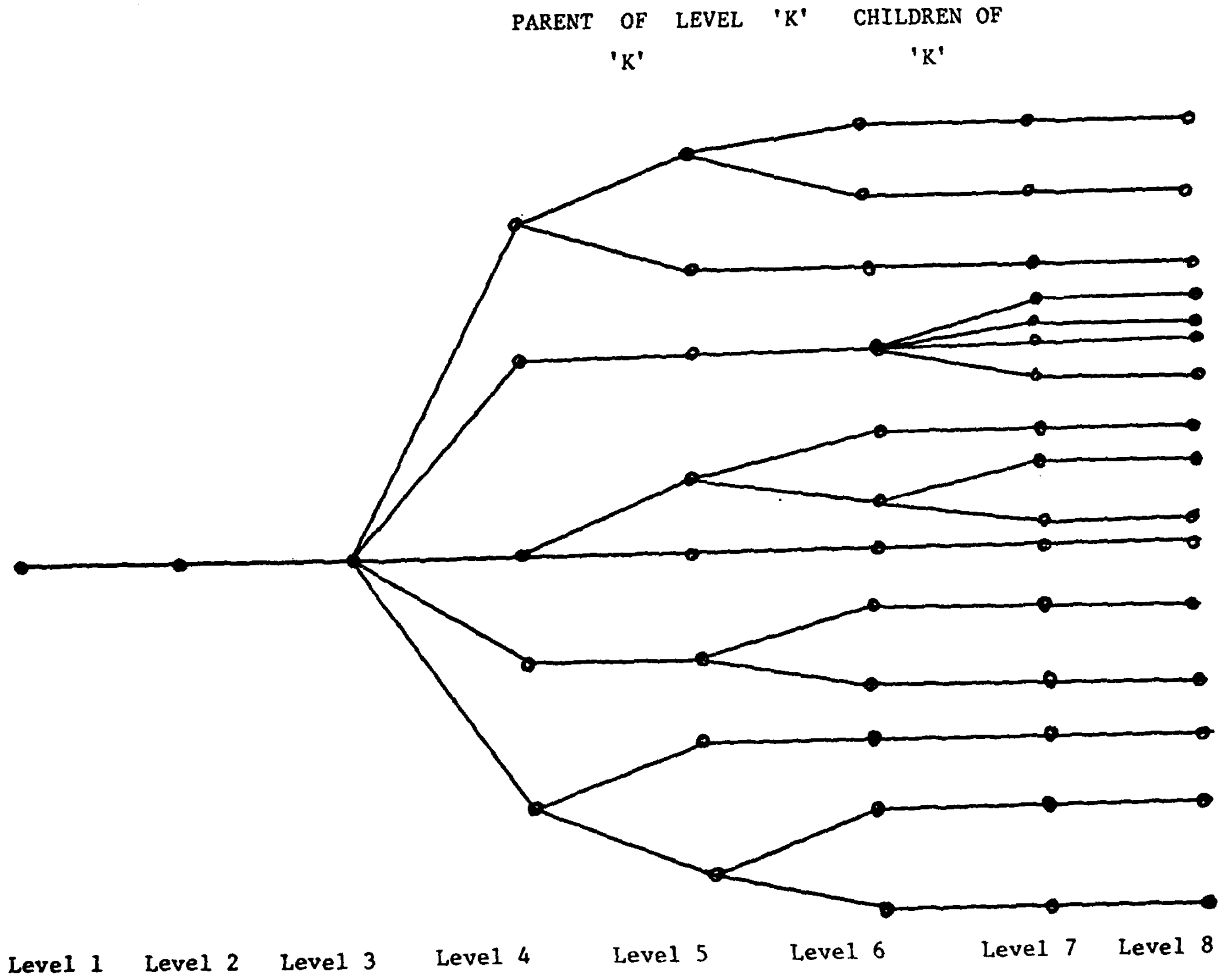


Figure 5. Schematic Diagram of a Very Simple Decision Tree

Because of the structural similarity of these decision trees to "family" trees, it is convenient to refer to nodes in terms of family relationships. Thus, the nodes in level K+1 are called the "Children" of the nodes in level K; nodes in level K-1 are the "Parents" of level K nodes; and nodes in level K-2 are the "Grandparents."

Basic program operation depends on the fact that nodes may be referred to:

- By the node designation discussed above, and
- By a node index giving the order of the node in the input file.

Efficient searching through the tree is permitted by the following relationships among the designations of the nodes in adjoining levels:

$$\text{Parent Node} = \text{Child Node} / 10 \qquad (1)$$

$$\text{Child Node} = \text{Parent Node} \times 10 + j \qquad (2)$$

where j = 1,... number of branches at Parent Node.

Equation (1) depends on the fact that, in FORTRAN, integer division loses any non-integer part. For example (221/10) = (222/10) = (223/10) = 22. Thus, all Brother Nodes produce the same Parent Node.

The node "index" is used as the index of arrays in which node attributes are stored--for example, NBR (number of branches at the Ith node), NOD (designation of the Ith node), LIN (number of lines in the Ith node's message). Thus, the number of branches at the Ith node is given by NBR (I).

File Structure

Information for each of the 52 networks is organized by node in a separate file for each. All data for a single node are stored on consecutive lines in the file. The following information is retained in the file for each node:

- Node designation - a set of alpha-numeric characters (up to 16) that identify both the tree and the node within the tree.
- Node level - the location in the network for a specific node.
- Number of branches at the node.
- Lines of text - each line is limited to 32 characters, but there is no limit to the number of lines per node (subject to the general limitation of 1500 lines for the entire tree).
- GO TO lines - pointers to related nodes.

A listing of a few lines from one of the tree files is shown in Table 3. This file has the following format:

Type of line	Columns	Variables
Node	1	Node level
Designation	3	Number of branches
Line	2-18	Node designation
Text Line	1	Blank if an additional line of text or a GO TO line follows: / if this is the last line and there are no GO TO lines.
	2-33	Text
GO TO Line	1	Line identifier: Must be *. Any text: for example, *GO TO P010204121.

PROGRAM DESCRIPTION

NET1 is made up of a main control program and five major subroutines: SF, SB, SR, CHILD, and PRINT. Flow diagrams showing the operations carried out by these programs are given in Figures 6-11.

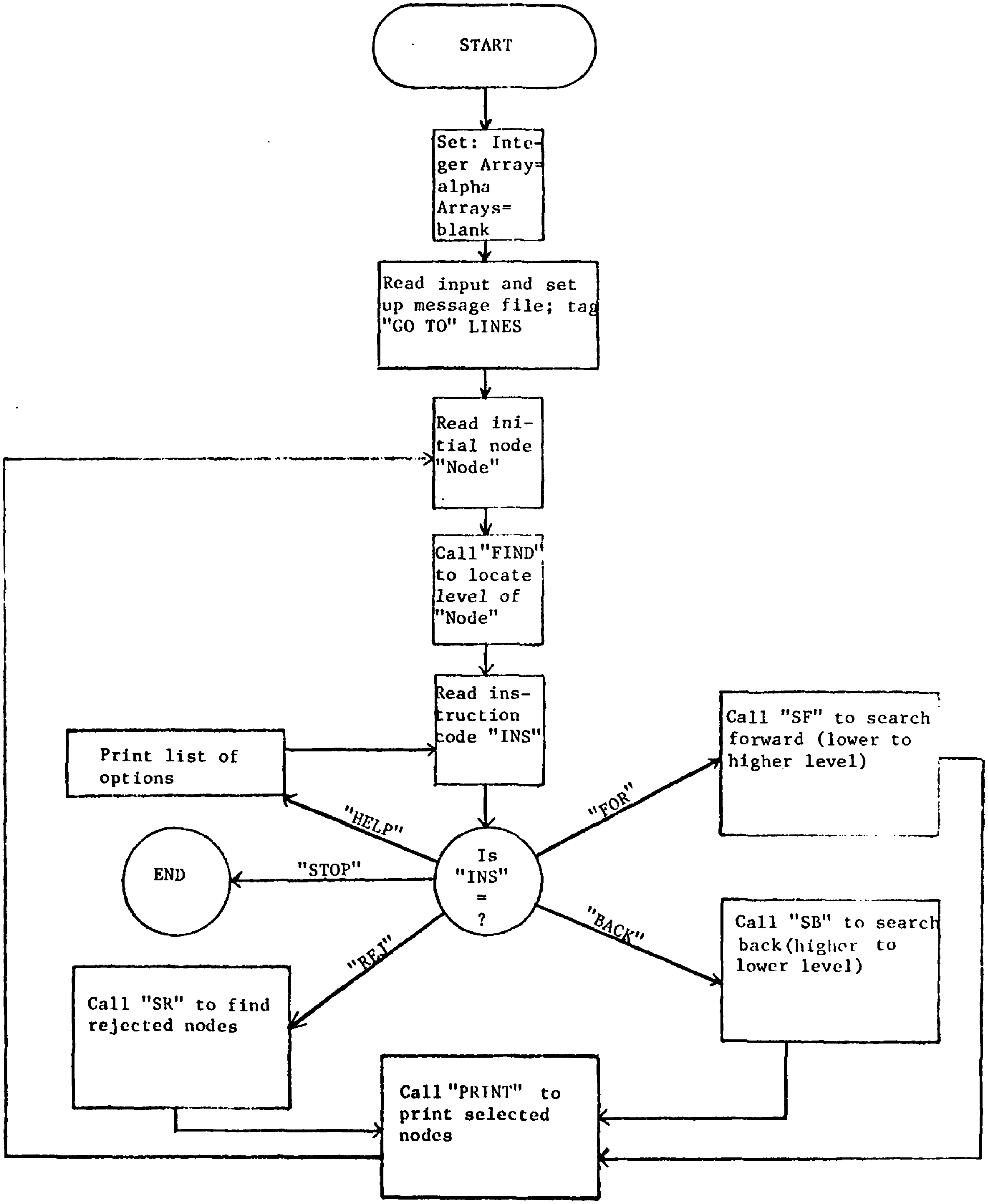
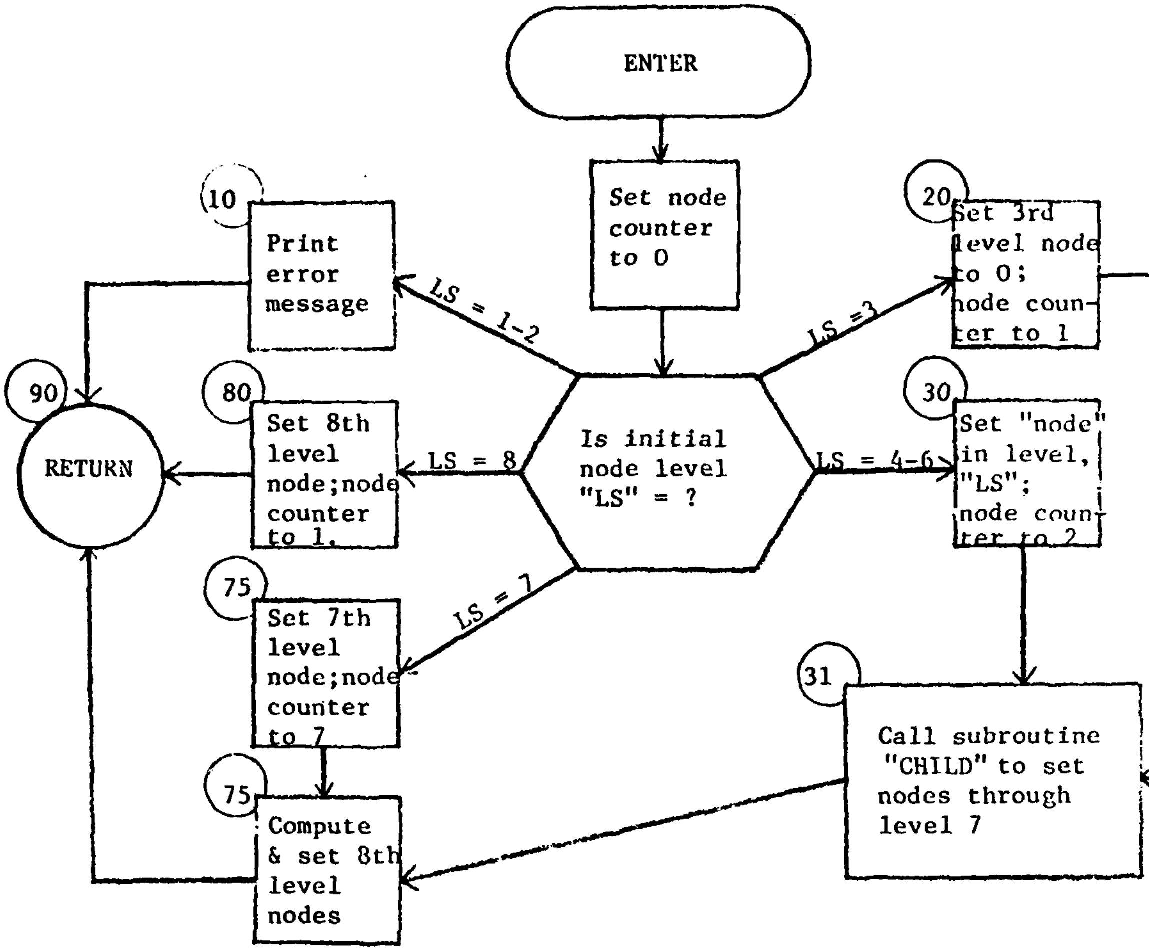

START
Set: Integer Array= alpha Arrays= blank
Read input and set up message file; tag "GO TO" LINES
Read initial node "Node"
Call "FIND" to locate level of "Node"
Read instruction code "INS"
Print list of options
Is "INS" = ?
"HELP"
"STOP"
END
"FOR"
Call "SF" to search forward (lower to higher level)
"BACK"
Call "SB" to search back (higher to lower level)
"REJ"
Call "SR" to find rejected nodes
Call "PRINT" to print selected nodes

Figure 6. NET1 - Main Program

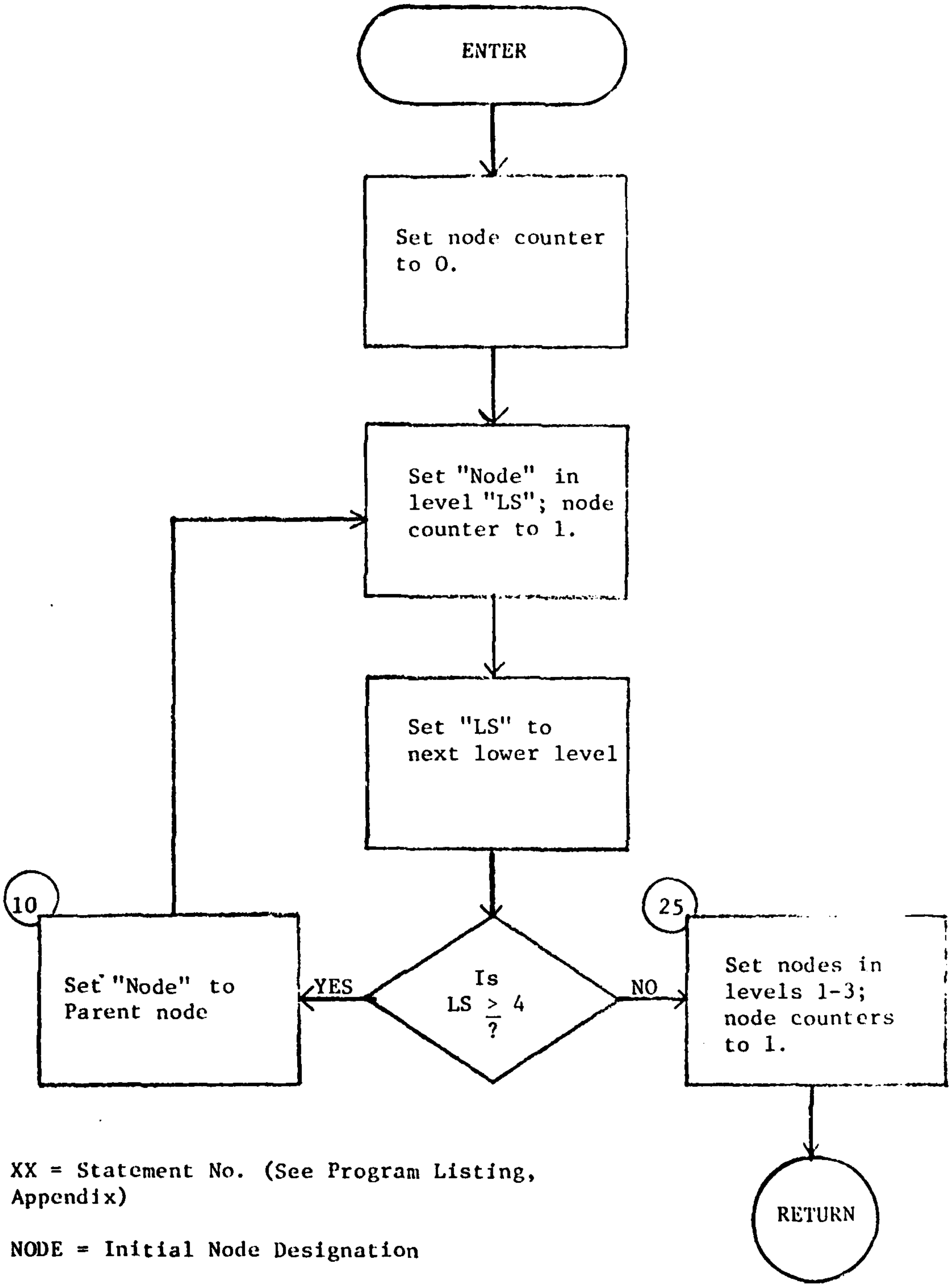

ENTER
Set node counter to 0
10
Print error message
LS = 1-2
20
Set 3rd level node to 0; node counter to 1
LS =3
Is initial node level "LS" = ?
90
RETURN
80
Set 8th level node;node counter to 1.
LS = 8
30
Set "node" in level, "LS"; node counter to 2
LS = 4-6
75
Set 7th level node;node counter to 7
LS = 7
31
Call subroutine "CHILD" to set nodes through level 7
75
Compute & set 8th level nodes

(XX) XX= Statement No. (See Program Listing, Appendix)

NODE - Initial Node Designation

LS - Initial Node Level

Figure 7. NET1: Subroutine SF (LLX, NEV, NODE, LS)

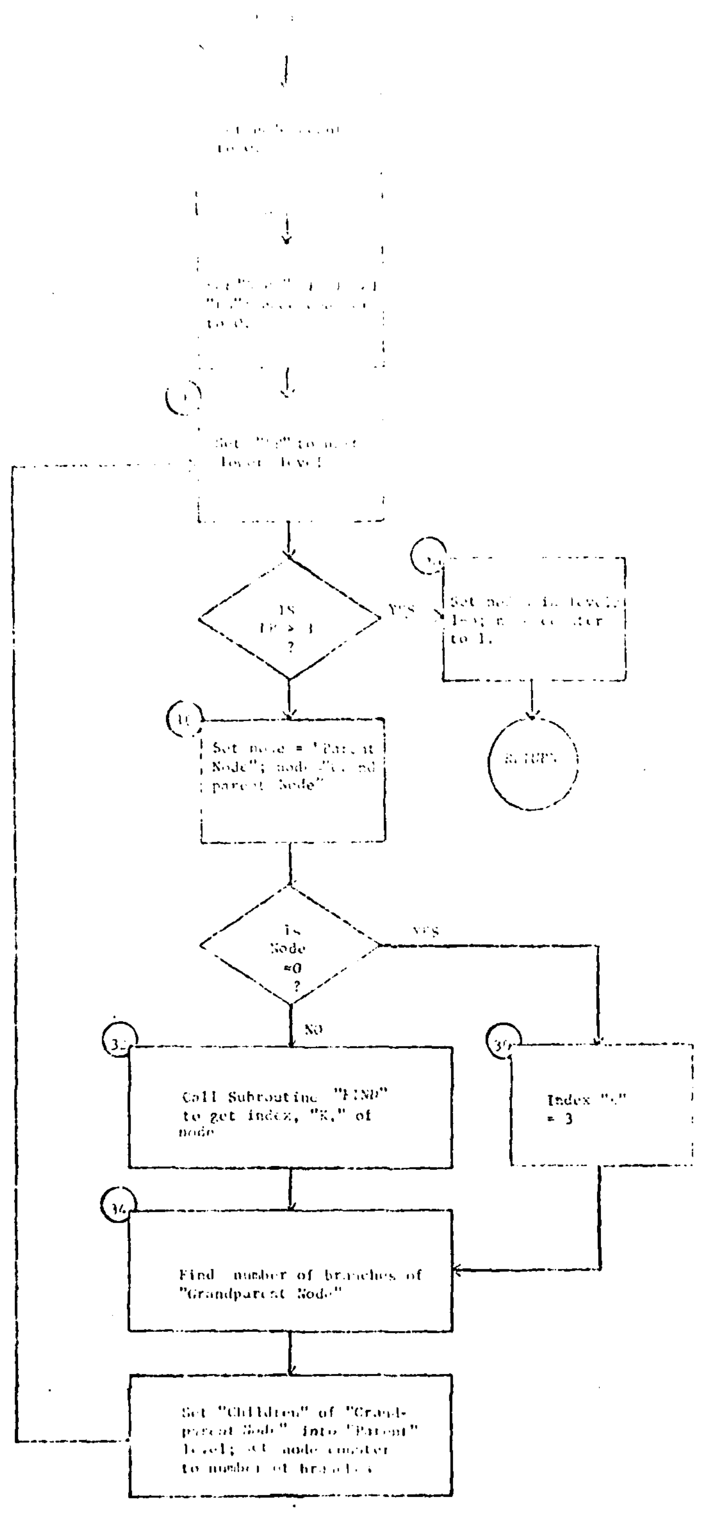

ENTER
Set node counter to 0.
Set "Node" in level "LS"; node counter to 1.
Set "LS" to next lower level
Is LS > 4 ?
YES
NO
10
Set "Node" to Parent node
25
Set nodes in levels 1-3; node counters to 1.
RETURN

XX = Statement No. (See Program Listing, Appendix)

NODE = Initial Node Designation

LS = Initial Node Level

Figure 8. NET1: Subroutine SB



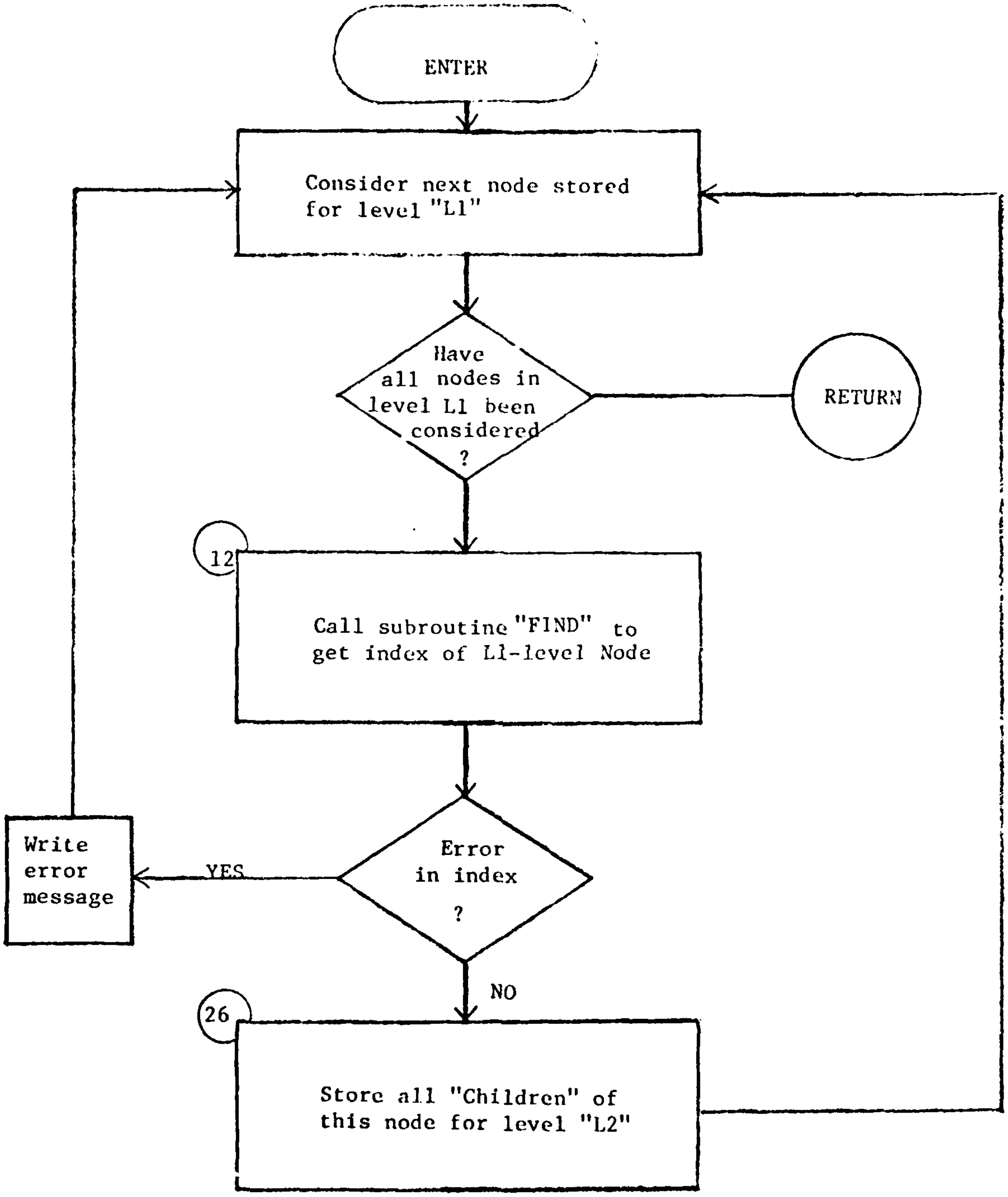
Set node counter to 0.
Set "LS" to next lower level
IS LS > 1 ?
YES
Set nodes in level 1; node counter to 1.
RETURN
Set node = "Parent Node"; node "Grand parent Node"
Is Node =0 ?
YES
NO
32
Call Subroutine "FIND" to get index, "K," of node
39
Index "K" = 3
34
Find number of branches of "Grandparent Node"
Set "Children" of "Grandparent Node" into "Parent" level; set node counter to number of branches

XX = Statement No. (See Program Listings Appendix)

NODE = Initial Node Designation

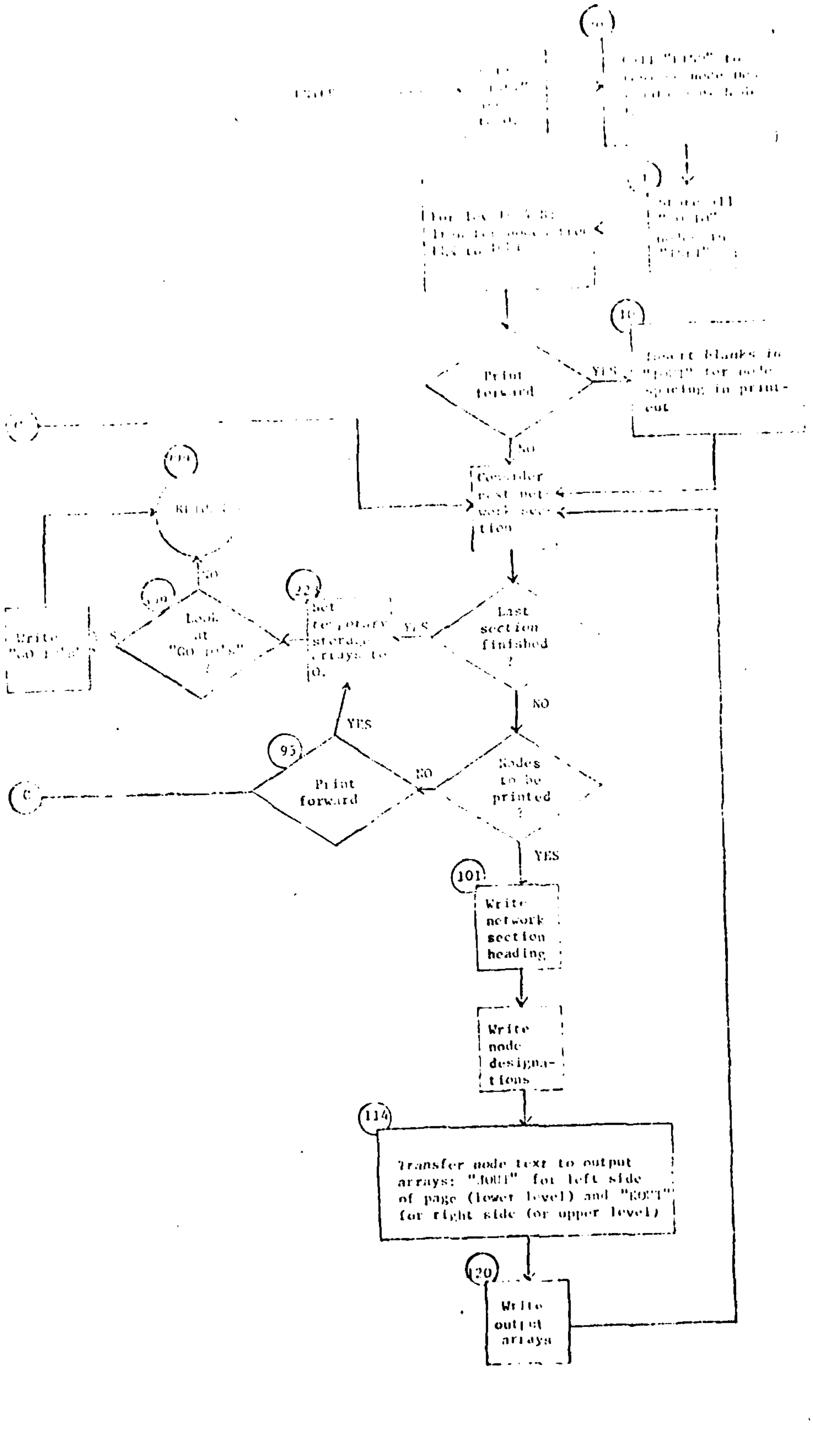
LS = Initial Node Level

Figure 9. RETI: Subroutine SB


ENTER
Consider next node stored for level "L1"
Have all nodes in level L1 been considered ?
RETURN
12
Call subroutine "FIND" to get index of L1-level Node
Error in index ?
YES
Write error message
NO
26
Store all "Children" of this node for level "L2"

XX XX = Statement No. (See Program Listing, Appendix)

Figure 10. NET1: Subroutine CHILD


10
Insert blanks in "port" for node spacing in print-out
YES
Print forward
NO
C
Consider next network section
RETURN
NO
Write "GO TO's"
YES
Look at "GO TO's" ?
22
Set temporary storage arrays to 0.
YES
Last section finished ?
NO
95
YES
C
Print forward
NO
Nodes to be printed ?
YES
101
Write network section heading
Write node designations
114
Transfer node text to output arrays: "JOUT" for left side of page (lower level) and "KOUT" for right side (or upper level)
120
Write output arrays

(XXX) XXX = Statement No. (See Program Listing Appendix)

Figure 11. NET1: Subroutine PRINT

TABLE 3

Sample of the NET1 Input: A Portion of One of the Tree Files

```
71T010201432.                               Node designation line
 LDC'S REJECT US OFFER FOR                  Text line
 TECHNICAL ASSISTANCE IN
 FAVOR OF EUROPEAN TIES.
*GO TO S0105021311.                         GO TO line
*GO TO S010502412.
71T0102014321.
 EUROPE ACCEPTS US PRESENCE IN
 LDC'S
*GO TO E01040142.
71T0102014322.
 EUROPE REJECTS US PRESENCE IN
 LDC'S COMPETES WITH US FOR
 LDC MARKETS.
*GO TO T010201431.
71T0102014331.
/NO THIRD ORDER EFFECT.
71T0102014341.
/NO THIRD ORDER EFFECT
71T0102015111.
/NO THIRD ORDER EFFECT.
71T0102015121.
 LDC'S HOLD RAW MATERIALS TO
 BARTER FOR TECHNOLOGY
 TRANSFER PROGRAMS.
*GO TO T01020152.
71T0102015122.
 LDC'S REJECT TECHNOLOGY
 FOR MORE CONVENTIONAL
 TYPES OF AID" POLITICAL,
/ECONOMIC, MILITARY.
71T0102015211.
 US FORCED TO LOOK ELSE-
 WHERE FOR RAW MATERIALS
 STOPPED IN RESPONSE TO US
/AID CUTOFF.
71T0102015221.
 LDC'S BREAK AWAY FROM
 EUROPE-SELL RAW
 MATERIALS TO HIGHEST
/TECHNOLOGICAL BIDDER.
80T0102014223.H041
 (-) RATE OF CHANGE OF TECH-
 NOLOGY DECREASED.  HIGH
 TECHNOLOGY FIRMS RELYING
 ON EXPORTS ARE FORCED OUT OF
/BUSINESS.
```

In addition to these programs, two minor utility subroutines, FIND and LEVEL, are included to determine the index and levei, respectively, for a given node. Definitions of the variables are given in Table 4.

TABLE 4

Definitions of the NET1 Variables

Variable Name	Definition
Arrays	
NBR(I)	Number of branches at the Ith node.
NOD(I)	Designation of the Ith node.
LOC(I)	Location in the message file of the first line of the message for the Ith node
LIN(I)	Number of lines in the Ith node's message
NOF(I)	Final part of the node designation
GTO(I)	Number of GO TO lines for this node
MES(L,K)	Text file (L=1,8 - Text /L=9 - tag for GO TO line
ISET(I)	Temporary storage for nodes with GO TO lines
NEV(L)	Number of nodes in the Lth level selected for printing
NST(L)	Total number of nodes in the Lth level
LLX(L,I)	Temporary storage for the designations of level L nodes selected for printing
LL	Powers of 10 used in identifying node levels
II	Temporary storage for one message line during the input phase
BUFFER	Required by the computer system for subroutine ATTACH
Single Variables	
WORDS	Number of (computer) words in a single message line
MAXLN	Maximum number of lines in the message for any node in a given tree (computed by the program)
NNOD	Maximum number of nodes in any single tree
NLEV	Number of levels in a tree (currently set at 8)
NLLX	Maximum number of nodes per level that could be selected during any single search
MMES	Maximum number of lines for all messages for a single tree

NET2: A BATCH MODE EQUIVALENCE SEARCH PROGRAM

PROGRAMMING PHILOSOPHY

The purpose of this batch program is to identify nodes with similar characteristics across the entire set of trees. The input file for NET2 is the Node Information File (NIF) which contains the following information for each node:

(1) Node designation

(2) Actor

(3) Event

(4) Target

(5) JCS geographic region

(6) Substantive topic

All nodes for which variables (2)-(6) are the same are considered to belong to the same "group." As each record is read, the group to which the node belongs is identified and stored with the other node attributes. This group identification is carried out by the following procedure:

Group 1 is defined by the attributes of the first node in the data file. If the second node is equivalent to Node 1, it is placed in Group 1; otherwise, its attributes define Group 2. If the third node is equivalent to either of the first two nodes, it is placed in the first or second group; otherwise, its attributes define Group 3.

When all of the nodes have been read, they are printed by groups. A pass is made through the list of nodes, and the nodes whose designations start with E are selected. Each E node serves as a "key" node (that is, it is printed at the left side of the page) and the designations of all nodes equivalent to it are printed to the right. Thus, each equivalent group is printed once for each member of the group. When all E nodes have been used as key nodes, the process is repeated with all M nodes, then P, S, and T nodes. (The order is controlled by the order of the values of the control variable, KON--set in a data statement. See the program listing in Appendix II.)

FILE STRUCTURE

Table 5 shows a listing of a few lines of the Node Information File. This file has the following format.[3]

Columns	Variables
1-11	Node designation
13-15	First Actor code (what country initiated the action)
16-18	Second Actor code
19-21	Action node (what the action was)
22-24	First Target code (toward what country or group was the action directed)
25-27	Second Target code
28	Region code (the geographic location of the action)
29-31	Substantive topic code (the subject topic of the exchange).

PROGRAM DESCRIPTION

NET2 consists of a main control program only. The flow diagram in Figure 12 indicates the operations carried out by the program. Definitions of the principal variables are given in Table 6.

[3] In the present program only parts of a larger data file are read. Thus, the actual data file contains information on two actors, two targets, and a three-digit code for action and substantive topic. Only one actor, one target, and two digits of the action and substantive topic codes are currently used in the search.

TABLE 5

Sample Listing of the Node Information File (NIF)

Node Designation / First Actor Code / Second Actor Code / Action Code / First Target Code / Second Target Code / JCS Region Code / Substantive Topic Code

```
F0101013211 7400000609970000101
F0101013221 9970000117400000138
F0101013221 7400000119970000133
F0101013222 9970002387400000133
F0101013222 7400002389970000133
F0101013222 7400000619950000180
F0101013231 7400000619950000180
F0101013232 7400002219970000138
F0101013311 7400002219970000100
F0101013312 7400000329970000147
F0101013312 7400001719950000130
F0101013321 7400001420029970180
F0101013322 7400001719970020107
F0101013331 7400001330020000107
F0101013332 7400002210020000100
F0101014121 3970001659950003180
F0101014122 3970001659950003147
F0101014122 3970001659950003180
F0101014131 3970001710029970107
F0101014132 3970001769970000103
F0101014133 3970001760020000103
F0101014141 3970001050020000180
F0101014142 3970002020020000180
F0101014221 3970001659950003147
F0101014221 3970001659950003180
F0101014222 3970000143650000140
F0101014223 3970002020020000180
F0101014321 3970002150020000180
F0101014322 3970000510020000137
F0101014331 3970002150020000180
F0101014332 3970000510020000100
F0101014411 0020002403970000100
F0101014412 0020000513970000100
F0101014412 0023972089950000130
               ↑↑↑  ↑   ↑↑↑   ↑
```

The (↑) designates columns not currently read by the program.

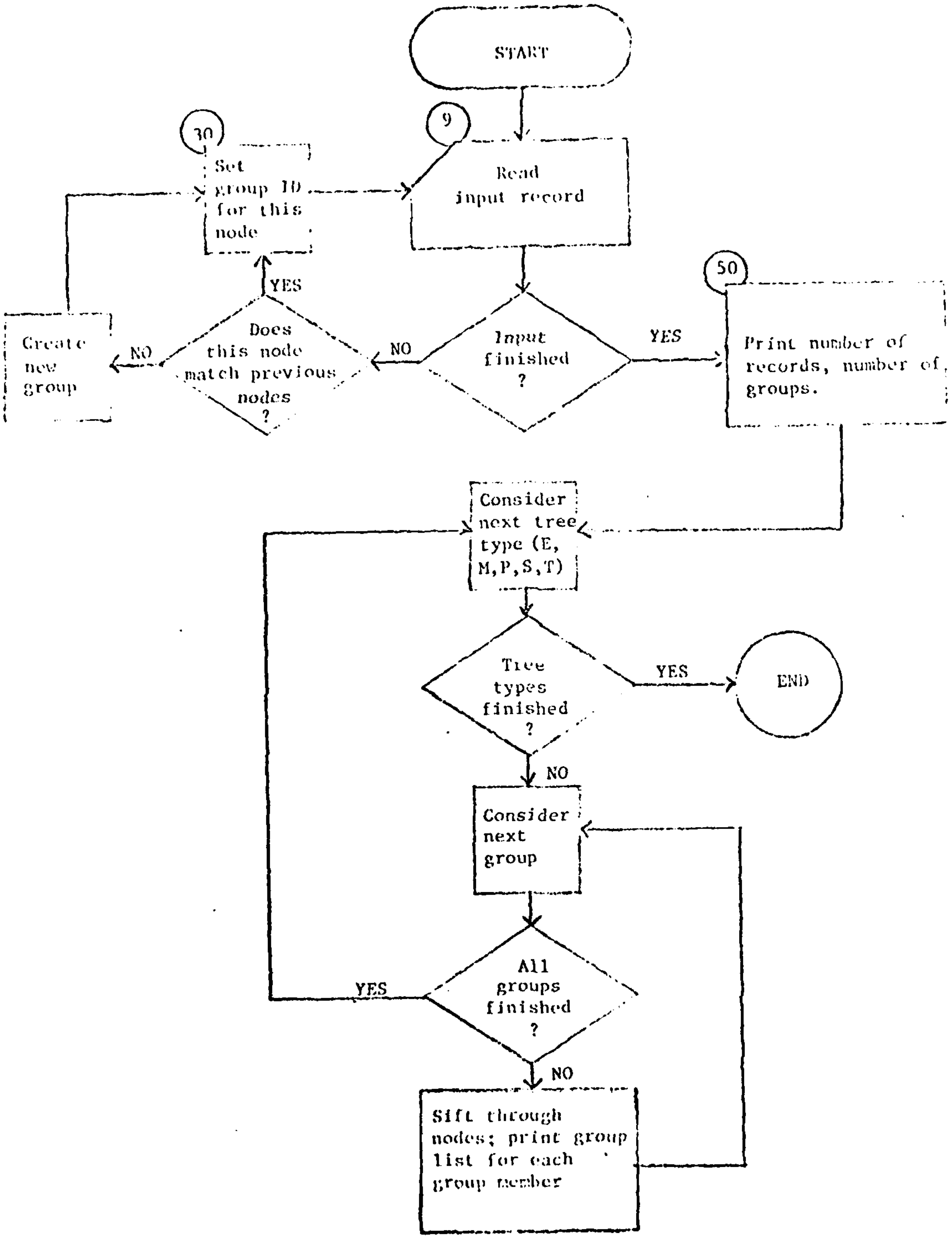

START
30
Set group ID for this node
9
Read input record
50
Does this node match previous nodes ?
NO
YES
Create new group
Input finished ?
NO
YES
Print number of records, number of groups.
Consider next tree type (E, M,P,S,T)
Tree types finished ?
YES
END
NO
Consider next group
All groups finished ?
YES
NO
Sift through nodes; print group list for each group member

Figure 12. Program NET2

TABLE 6

Definitions of the NET2 Variables

Variables Names	Definition
Arrays	
NN(I,J)	Designation of the Ith node (j=1,5; computer words are required)
II(I)	(Packed) word containing actor, event, and target codes for the Ith node
LR(I)	Geographic region code for the Ith node
LA(I)	Substantive topic code for the Ith node
IG(I)	Equivalence group of the Ith node
KON(J)	Letter code for the Jth tree type (J=E,M,P,S, and T)
N(K)	Input buffer (K=5)

NET3: AN INTERACTIVE EQUIVALENCE SEARCH PROGRAM

PROGRAMMING PHILOSOPHY

This interactive program reads the Node Information File (NIF)--which was described for NET2--and allows the user to print information on nodes selected by one or more of their attributes. Thus, a user may ask to see all node entries with actor code 365, or all node entries with substantive topic code 09, or all entries with actor code 365 and substantive area 09.

The program reads the user's responses to questions designed to elicit his selection criteria and sets up a selection array, each word of which contains either the user's selected code for the corresponding attribute or the value -1 indicating that this variable is not used in the selection process.

PROGRAM DESCRIPTION

NET3 consists of a main control program and a utility subroutine to convert alpha-characters to integers. The flow diagrams in Figures 13A and 13B indicate the operations carried out by the program. Definitions of the arrays are given in Table 7.

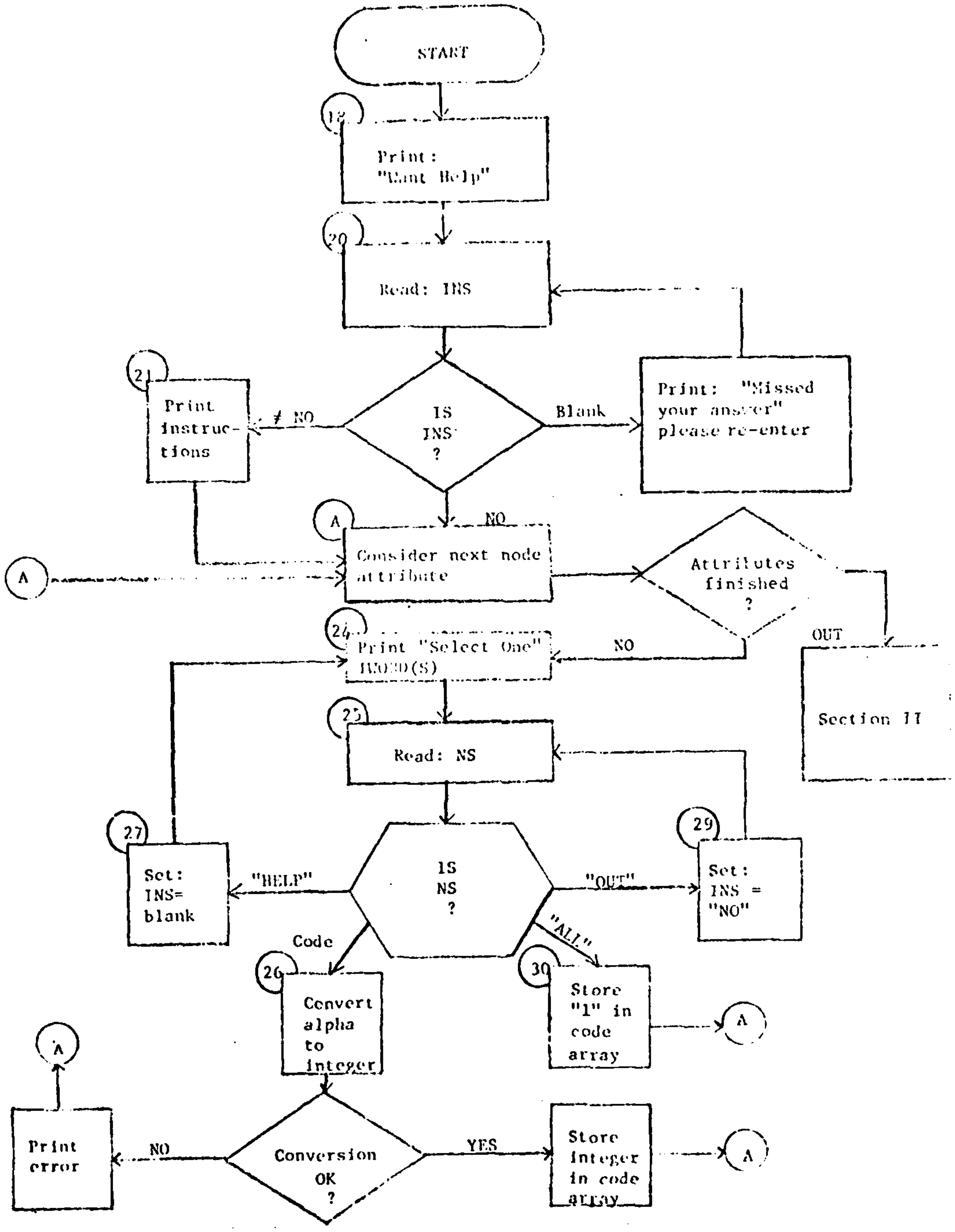


Figure 13A. NET3: Section 1: Read User's Selection Code

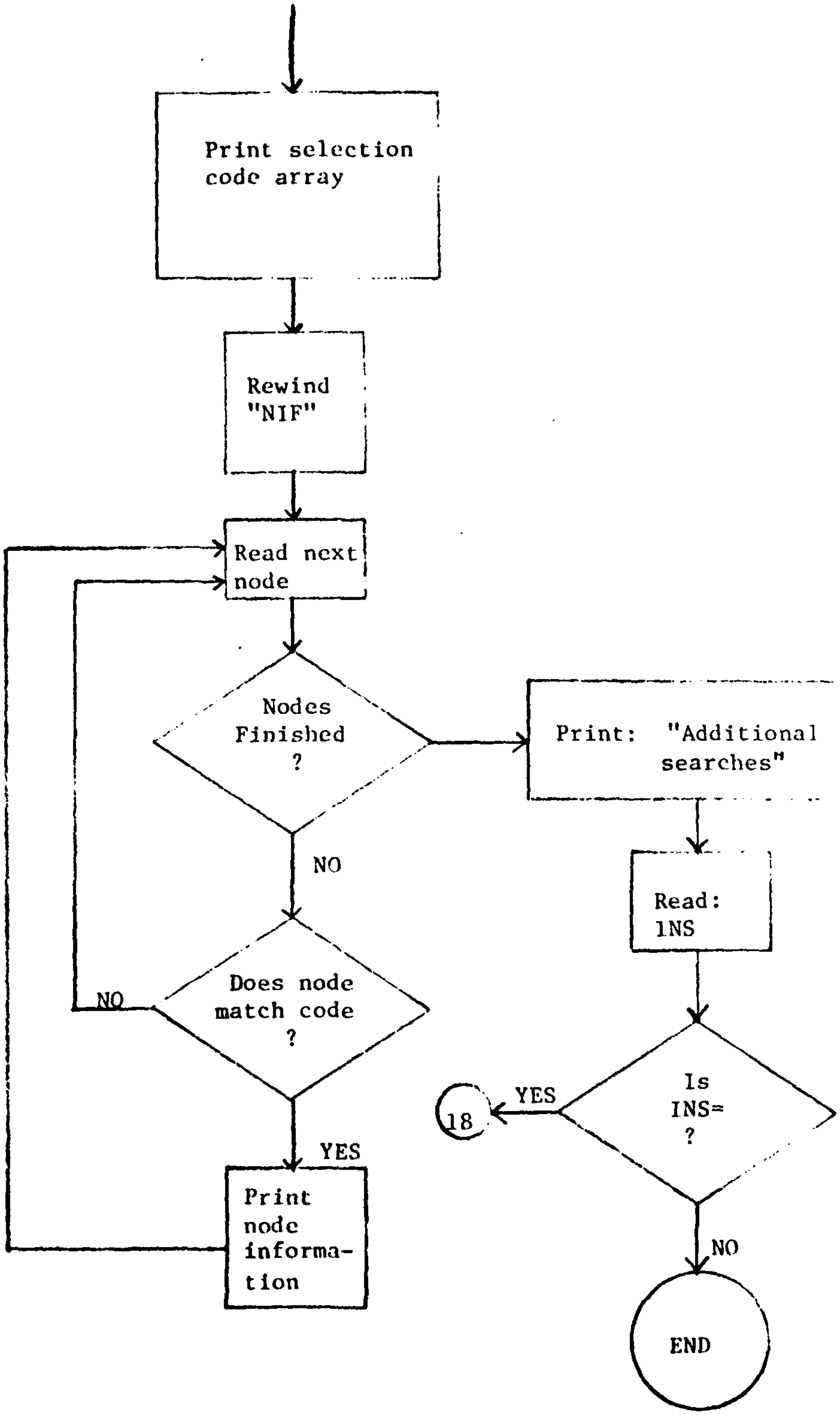


Figure 13B. NET3: Section II: Search Node Information (NIF) for Matches to Selection Code.

TABLE 7

Definitions of the NET3 Variables

Variable Name	Definition	Format
Arrays		
IWORD(K) (K=1,5)	List of the selection code	A4
IXX(I,1)	Actor code for the Ith node	I3
IXX(I,2)	Action code for the Ith node	I2
IXX(I,3)	Target code for the Ith node	I3
IXX(I,4)	Geographic region code for the Ith node	II
IXX(1,5)	Substantive topic code for the Ith node	I2
NN(I,K) (K=1,3)	Designation of the Ith node (3 variables)	A4,I5,A4
BUFFER	System variable required for file handling	

NET4: A NETWORK SUMMARY PROGRAM

PROGRAMMING PHILOSOPHY

NET4 allows the user to search the Tree Summary File (TSF) for information across the entire set of trees.

The user may retrieve and print information for trees selected by any of eight variables. In addition to displaying sections of the TSF, the command set allows the user to modify file entries and to add new ones (although these latter functions may also be carried out with the USAWC computer system's EDIT command).

FILE STRUCTURE

The Tree Summary File contains the following information for each of the networks in the FORECAST 90 system:

- Tree number
- Author of the tree
- Date of tree creation
- Date of last tree modification
- Total number of nodes in the tree
- Actor code
- Substantive topic code
- Region code
- One line of text describing the subject matter.[4]

[4] More complete information on the catalytic events used to build the networks is given in Appendix I of the "User's Manual for the FORECAST 90 Computer Programs." Information on the actor, substantive topic and region codes is given in Appendices III-VII of that manual.

TABLE 5

Sample Listing of the Tree Summary File (TSF)

Tree Number / Tree Author / Date Created (Yr., Month, Day) / Date Modified (Yr., Month, Day) / Number of Nodes / Actor Code / Substantive Area Code / Region Code

```
EØ1Ø1Ø1  CACI 75Ø3Ø7 75Ø7Ø7 1Ø1 691 14 Ø5
EØ1Ø1Ø1  OIL EMBARGO AGAINST WESTERN STATES BY MIDDLE EASTERN COUNTRIES.
EØ1Ø1Ø2  CACI 75Ø3Ø7 75Ø7Ø7 133 Ø22 14 Ø1
EØ1Ø1Ø2  US GRAIN EMBARGO.
EØ1Ø1Ø3  CACI 75Ø3Ø7 75Ø7Ø7 154 397 14 Ø3
EØ1Ø1Ø3  EEC ATTEMPTS TO BECOME RESOURCE INDEPENDENT BY TRADE PACTS WITH LDCS.
EØ1Ø1Ø4  CACI 75Ø3Ø7 75Ø7Ø7  84 596 14 Ø2
EØ1Ø1Ø4  CARTELIZATION OF COPPER.
EØ1Ø2Ø1  CACI 75Ø3Ø7 75Ø7Ø7 113 74Ø 14 Ø7
EØ1Ø2Ø1  JAPAN-USSR TRADE/AID PACTS TO DEVELOP SOVIET ENERGY SOURCES.
EØ1Ø3Ø1  CACI 75Ø3Ø7 75Ø7Ø7 17Ø 397 14 Ø3
EØ1Ø3Ø1  EUROPEAN ECONOMIC INTEGRATION.
EØ1Ø3Ø2  CACI 75Ø3Ø7 75Ø7Ø7  99 ØØ2 Ø2 Ø1
EØ1Ø3Ø2  SEVERE ECONOMIC RECESSION OCCURS IN THE US.
EØ1Ø4Ø1  CACI 75Ø3Ø7 75Ø7Ø7 133 997 12 ØØ
EØ1Ø4Ø1  LDC'S PRESSURE DEVELOPED COUNTRIES FOR NEW AID SYSTEM.
EØ1Ø5Ø1  CACI 75Ø3Ø7 75Ø7Ø7 1Ø1 397 14 Ø3
EØ1Ø5Ø1  EEC BREAKS DOWN FROM STRAINS OVER RESOURCE AVAILABILITY.
MØ1Ø1Ø1  CACI 75Ø3Ø7 75Ø7Ø7  7Ø 365 17 Ø4
MØ1Ø1Ø1  USSR EXPANDS STRATEGIC WEAPONS, SEEKS FIRST STRIKE CAPABILITY.
MØ1Ø2Ø1  CACI 75Ø3Ø7 75Ø7Ø7 137 ØØ2 Ø7 Ø3
MØ1Ø2Ø1  MBFR UNSUCCESSFUL.  CONGRESS MANDATES 5Ø,ØØØ TROOP WITHDRAWAL.
MØ1Ø3Ø1  CACI 75Ø3Ø7 75Ø7Ø7  54 ØØ2 Ø7 ØØ
MØ1Ø3Ø1  US LOSES BASE/TRANSIT RIGHTS IN SOUTHERN EUROPE AND MIDDLE EAST.
MØ1Ø3Ø2  CACI 75Ø3Ø7 75Ø7Ø7 172 ØØ2 Ø7 Ø7
MØ1Ø3Ø2  US LOSES BASE/TRANSIT RIGHTS IN JAPAN AND OKINAWA.
MØ1Ø3Ø3  CACI 75Ø3Ø7 75Ø7Ø7 17Ø ØØ2 Ø7 Ø7
MØ1Ø3Ø3  US LOSES BASE/TRANSIT RIGHTS IN THAILAND, TAIWAN, AND PHILIPPINES.
MØ1Ø3Ø4  CACI 75Ø3Ø7 75Ø7Ø7 1ØØ ØØ2 17 Ø1
MØ1Ø3Ø4  US NAVAL CAPABILITY INCREASED WITH SURFACE EFFECT SHIPS.
MØ1Ø3Ø5  CACI 75Ø3Ø7 75Ø7Ø7  99 ØØ2 17 Ø1
MØ1Ø3Ø5  IMPROVED LOGISTICAL CAPABILITIES PERMIT RAPID US BASING ALMOST ANYWHER
```

Table 8 is a listing of a few lines of the Tree Summary File. This file has the following format:

Type of Line	Columns	Variables
Tree Designation Line	1- 8	Tree number
	10-13	Author
	15-20	Date created (year, month, day)
	22-27	Date modified (year, month, day)
	29-31	Number of nodes in the tree
	33-35	Actor code
	37-38	Substantive topic code
	40-41	Region code
Text Line	1-80	Up to 80 characters of text to describe the focus of the network.

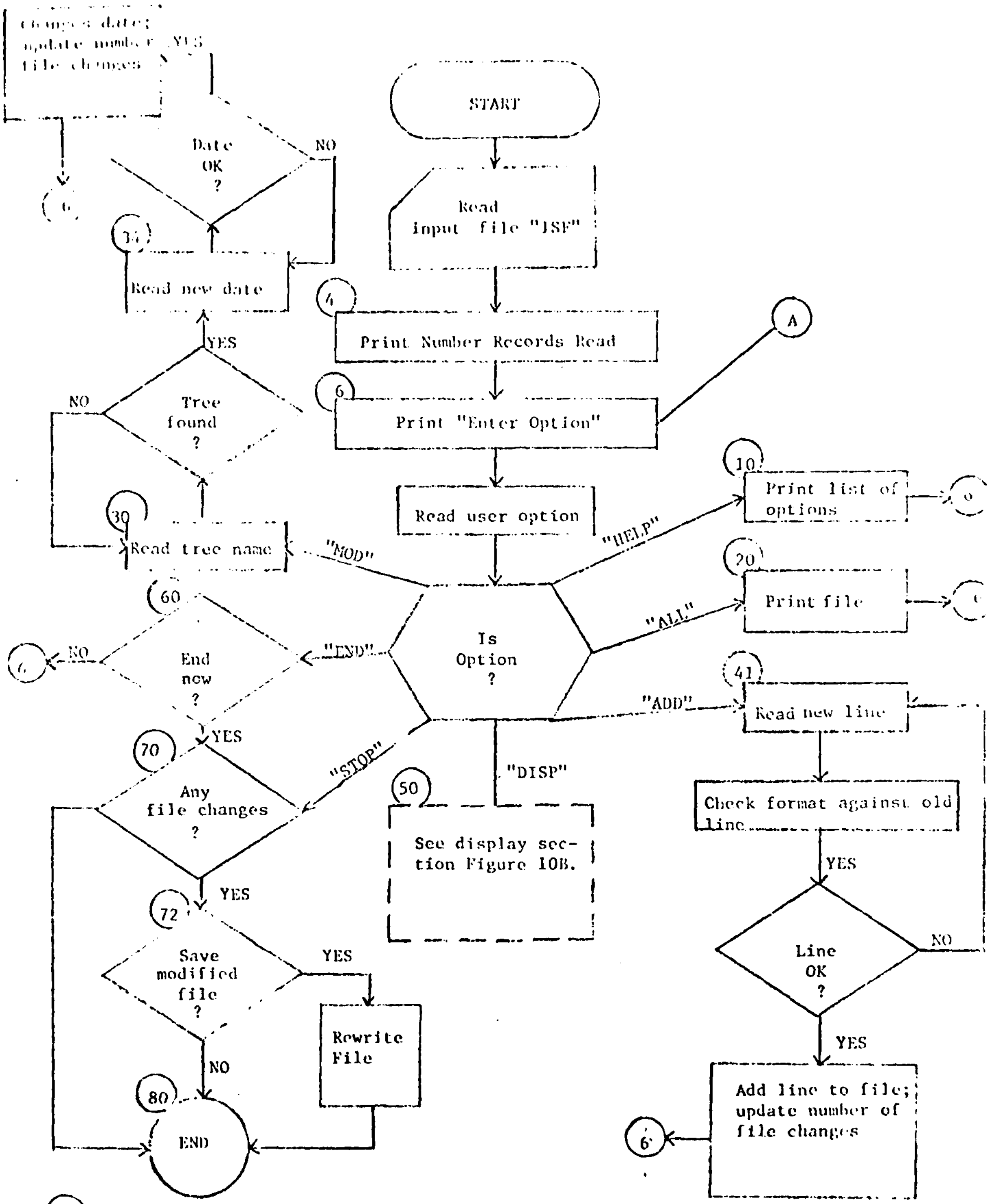
PROGRAM DESCRIPTION

NET4 consists of a main program only. An operational level flow diagram is given in Figures 14A and 14B. Definitions of the variables are given in Table 9.

TABLE 9

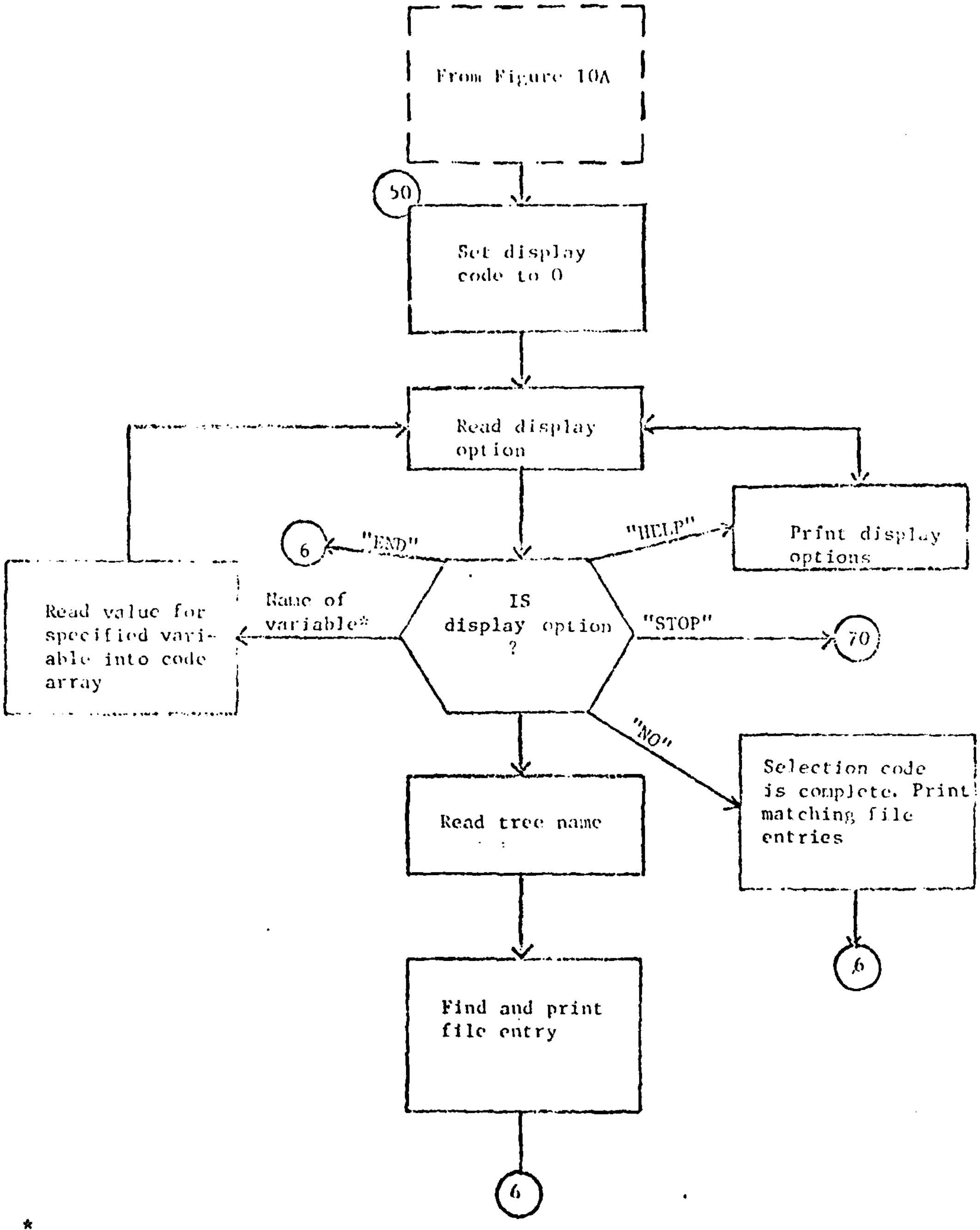
Definitions of the NET4 Variables

Variable Name	Definition	Format
Arrays		
IX(K,L) L=1,2	Tree name	2A4
IX(K,3)	Author of the tree	A4
IX(K,4)	Date tree was created	I6
IX(K,5)	Date of last tree modification	I6
IX(K,6)	Number of nodes in the tree	I3
IX(K,7)	Actor code	A3
IX(K,8)	Substantive topic code	A2
IX(K,9)	Geographic region code	A2
IX(K,L), L=10,29	Descriptive text	20A4
BUFFER	System variable required for file handling	



XX = Statement No. (See Program Listing in Appendix)

Figure 14A. Program NET4: Option "MOD," "END," "STOP," "HELP," "ALL," and "ADD."



* The descriptive variables by which file entries may be selected are: "AUTH," "DEC," "DEN," "ACT," "SUB," and "REG."

Figure 14b. Program NET4: Display Option.

GLOSSARY OF TERMS FOR THE NET PROGRAMS

Alpha-array — An array in which letters or other symbols are stored; it may be used in comparison statements but not in computations.

Message — The text, or descriptive material, accompanying a node (arranged in 32-character lines).

Node Designation — The set of letters and integers that identify a node; the name of the node.

Node Index — An integer specifying the order of the node in the input deck; used to index arrays containing node attributes.

Attribute — Numerical information about an entity; for example, the number of branches at a node is a node attribute.

Parent Node — The Parent Node of a node in, say, level K is the node in level K-1, directly connected to it. A given node may have only one parent.

Child Node — A child of a node in level K is a node in level K+1 directly connected to it. A Parent Node may have several children.

Appendix I; Annotated Program Listings for NET1

PROGRAM NET1

Purpose: To retrieve and print selected parts of a single tree.

MAIN CONTROL PROGRAM

Purpose: To process the input data describing the tree and to execute user-specified options.

```
      DIMENSION GTO(600),LIN(600),LOC(600),NBR(600),NOD(600),NOF(600)
     &,NEV(8),NST(8),LLX(8,90)
     &,MES(9,1500),BUFFER(380)
      COMMON GTO,LIN,LOC,NBR,NOD,NOF,MES,NST,LLX,NEV
     &,WORDS,NM1,NM2,NM3,MAXLN,IFO
      ASCII ANS,IW,IFILE(6),JW,NO
      DIMENSION LL(7),II(8),KS(5)
      INTEGER WORDS,GTO
      DATA NNOD,NLEV,NLLX,NMES/600,8,90,1500/
      DATA KS/3HFOR,3HBAC,3HREJ,3HSTO,3HHEL/
      DATA IFILE/"AWC1","SS/C","ACI/","    ","    ",";   "/
      DATA IAS,LANK/1H*,1H /,NO/2HNO/
```

Pre-processing phase. Integer arrays are set to 0; alpha arrays, to blanks.

```
      PRINT,"THE NET1 PROGRAM ALLOWS YOU TO PRINT"
      PRINT,"PORTIONS OF A SPECIFIED TREE"
      IFO=6
      PRINT,"OUTPUT AT TERMINAL "
      READ,ANS
      IF(ANS.EQ.NO)IFO = 10
      WORDS=8
      IW=0
      DO 1  K=1,NNOD
      NOF(K)=LANK
      NBR(K)=0
      NOD(K)=0
      LOC(K)=0
      LIN(K)=0
      GTO(K)=0
    1 CONTINUE
      MAXLN=0
      DO 2  K=1,NLEV
    2 NST(K)=0
      DO 4  K=1,NLLX
      DO 3  J=1,NLEV
    3 LLX(J,K)=0
    4 CONTINUE
      DO 6  K=1,NMES
      MES(9,K)=0
      DO 5  J=1,WORDS
    5 MES(J,K)=LANK
    6 CONTINUE
```

PROGRAM NET1

MAIN CONTROL PROGRAM (continued)

Powers of 10 are stored in array 'LL' for use in determining the level of a node from the numerical value of its designation. (See subroutine LEVEL.)

```
360        LL(1)=1
370        DO 8  K=1,6
380      8 LL(K+1)=10**K
```

The user is asked to enter the name of the tree he wishes to consider. After several counters are set to 0 and scratch file 02 is created, the specified tree is attached as file

```
390   181 PRINT,"ENTER TREE"
400       READ 182,IW,JW
402   182 FORMAT(A4,A3)
410       PRINT 182,IW,JW
420       PRINT,"OK "
430       READ,ANS
440       IF(ANS.EQ."NO")GO TO 181
450       IFILE(4)=IW
452       IFILE(5)=JW
460       IG=0
470       IN=0
480       MX=0
490       F=0.
500       IH=LANK
510       CALL CREATE(02,1000,0,ISTAT)
520       CALL ATTACH(01,IFILE,1,0,ISTAT,BUFFER)
```

Input phase. Data for nodes in the first three levels are read. It is from the node designations for the first level nodes that the initial seven characters of the node designatiors are read. These fields are skipped for the remaining nodes.

```
530   180 READ (1,411) N1,N2,NM1,NM2,NM3
540   411 FORMAT(2I1,A3,2A2)
550       GO TO 210
560   190 IF (IN-3)  180,191,191
```

The remaining input is read.

```
570   191 READ(1,200,END=220)N1,N2,F,IH
580   200 FORMAT(2I1,7X,F5.0,A4)
590   192 IF (N1-9) 210,220,220
```

IN is the index of the node currently being processed. Attributes of this node are stored. For level 8 nodes, the node designation must be computed. (On the input cards a level-8 node is identified by the characters--up to 4--following the decimal point in the node designation. Internal to the program, level 8 nodes follow the same pattern in their designations as the other nodes; for example a level 8 node will have one more integer than its level 7 Parent Node. Thus,

```
210 IN=IN+1
    NBR(IN)=N2
    IF (N1.EQ.8)  F=F*10.+1.
    NOD(IN)=F
    NOF(IN)=IH
    IF (IN-4)  208,208,213
208 LS=IN
    GO TO 218
213 CALL LEVEL(LL,NOD(IN),LS)
216 IF (LS-N1)  217,218,217
217 PRINT 203,N1,LS,NOD(IN)
203 FORMAT(1X,6H ERROR ,2I2,I8)
218 NST(LS)=NST(LS)+1
```

The texts for all of the nodes are stored in a single array, MES. The location in this array of the first line of text for node IN is stored in the array LOC. One line of text is read, counted, and stored.

```
    LOC(IN)=IM+1
211 READ(1,202,END=220)KODE,II
202 FORMAT(A1,8A4)
194 LIN(IN)=LIN(IN)+1
    IM=IM+1
    IF (IM.GT.1500) GO TO 219
    DO 212  K=1,WORDS
212 MES(K,IM)=II(K)
```

The local variable, KODE, contains the character found in the first column of the text card. The three allowed values and their meanings are:

(1)	blank	There are additional lines of text for this node.
(2)	/	This is the last line of text for this node and there are no GO TO nodes.
(3)	*	This line specifies a GO TO node.

Any character in the first column of a text card other than blank, /, or * is treated as a / .

KODE = blank ; reading of the text continues. MAXLIN is reset, if necessary.

```
810       IF (KODE.EQ.LANK) GO TO 211
820       IF (LIN(IN).GT.MAXLN) MAXLN=LIN(IN)
```

KODE ≠ blank ; this card must be either the last text card or the first GO TO line. If KODE ≠ * , this card is the last text card and control must be passed back to statement 190 to read the next node card.

```
830       IF (KODE.NE.IAS) GO TO 190
```

KODE = * ; this is a GO TO line. It is tagged as a GO TO line and counted.

```
840       MES(9,IM)=1
850       GTO(IN)=GTO(IN)+1
```

The next card is read. Now if KODE ≠ *, the card must be a new node card. It must be written to scratch file ('02') so that it may be reread in the correct format. Control is then returned to statement '192' so that the new node may be processed.

```
860       READ(1,202) KODE,II
870       IF (KODE.EQ.IAS)  GO TO 194
880       REWIND 2
890       WRITE(2,202) KODE,II
900       REWIND 2
910       READ(2,200) N1,N2,F,IH
920       GO TO 192
```

Reading of the input is ended when the end of file is encountered or when the message file is filled (it has a maximum of 1000 lines as currently dimensioned). The total number of nodes in the tree and the nodes per level are printed.

```
 930  219 PRINT,"MESSAGE FILE TOO LARGE"
 940  220 PRINT 300,IN,(NST(J),J=1,8)
```

```
950   300 FORMAT(27HINPUT FINISHED NO. NODES = ,I4/ 15H NODES/LEVEL :  ,
960       &8I5)
```

The next section writes the node designation, number of branches, and number of GO TO's for each node in the tree to scratch file 09. This information may be used in checking new trees. Currently, lines 961-966 are inactive. To reactivate this section, the letter C following the line number must be removed.

```
961C      CALL CREATE(09,500000,0,ISTAT)
962C      DO 884 K=1,IN
963C  884 WRITE (9,886)NOD(K),NOF(K),NBR(K),GTO(K)
964C  886 FORMAT(I7,A4,4(1X,I4))
965C      CALL APRINT(09,"AWC1SS.FRCST 90",9,"RA")
966C      CALL DETACH(09,ISTAT,BUFFER)
```

The first three words in array NOD are set to 0. The user is asked to enter the initial node he wishes to consider. He is allowed to enter the full node designation, although the program reads only the four integers following the first seven characters. Any character punched beyond the four integers indicates a level 8 node, whose designation internal to the program must be computed as in the initial data entry.

```
970       DO 207  L=1,3
980   207 NOD(L)=0
990   223 PRINT 225
1000   225 FORMAT(1X,18HENTER INITIAL NODE   )
1010   224 READ 226,F,IH
1020   226 FORMAT(7X,F5.0,A1)
1030       IF (IH.NE.LANK) F=F*10.+1.
1040       NODE=F
```

The level of the node is determined. (Currently, the initial node may not be in levels 1 or 2.)

```
1050       IF (NODE)  400,400,401
1060   400 LS=3
1070       GO TO 500
1080   401 CALL FIND(NODE,NOD,KK)
1090       IF (KK.NE.999) GO TO 402
1100       PRINT 227
1110   227 FORMAT(1X,30HERROR IN NODE. PLEASE RE-ENTER   )
1120       GO TO 224
1130   402 CALL LEVEL(LL,NODE,LS)
```

PROGRAM NET1

MAIN CONTROL PROGRAM (continued)

The user is asked to enter his option. His available choices (FOR,BAC,REJ,HEL, and STOP) were stored in array KS, which must be searched to find which he has selected. If his entry cannot be identified, the options are printed and he is given a second chance.

```
500 PRINT 601
601 FORMAT(1X,12HENTER OPTION   )
    READ 502, INS
502 FORMAT(A3)
    DO 504  K=1,5
    IF (INS.EQ.KS(K))  GO TO 506
504 CONTINUE
    GO TO 508
```

Program control is transferred to the user's option.

```
506 GO TO (510,520,530,999,508),K
```

The user has asked for instructions by entering HELP (or HEL, since only the first

---

```
508 PRINT 509
509 FORMAT(10H OPTIONS :  /31H FOR - PRINT HIGHER LEVEL NODES  /
   &31H BACK - PRINT LOWER LEVEL NODES/26HREJ - PRINT REJECTED NODES/
   &21H HELP - PRINT OPTIONS  / 31H STOP - HALT PROGRAM EXECUTION  )
    GO TO 500
```

The user's print options and their corresponding subroutines are:

FOR- to print forward- Subroutine SF

BAC- to print back-Subroutine SB

REF- to print rejected nodes- Subroutine SR

The subroutine matching the user's option is called. It will store the nodes to be printed in array LLX.

```
1280   510 CALL SF(LLX,NEV,NOD,NBR,NODE,LS)
1290       M=0
1300       GO TO 540
1310   520 CALL SB(LLX,NOD,NEV,LS,NODE)
1320       M=1
1330       GO TO 540
1340   530 CALL SR(LLX,NOD,NEV,LS,NODE,NBR)
1350       M=2
```

The nodes are printed. First, the temporary output file, 10, is created. Subroutine PRINT is called to write the output to 10. If the output has not already been printed at the terminal, then the AWC-supplied subroutine APRINT is called to print file 10, and the termporary file is detached. Program control is returned to statement 223 to allow the user to enter a new initial node. Note: The location of the CREATE and DETACH commands causes a new set of output (for example, a new job) for each user request. Alternatively, the program could be modified to write all output from a single program execution to file 10 before APRINT is called and the file detached. However, in case of an error in any one of the user requests, no output, even from correct requests, would be produced. The current procedure seems safer.

```
1360  540 CALL CREATE (IFQ,500000,0,ISTAT)
1370      CALL PRINT(M)
1375      IF(IFQ.EQ.6)GO TO 550
1380      CALL APRINT(IFQ,"AWCISS,FRCST 90",9,"RA")
1390  550 CALL DETACH(IFQ,ISTAT,BUFFER)
1400      GO TO 223
```

The input file 01 and the scratch file 02 are detached. Execution is halted.

```
1410   999 CALL DETACH(02,ISTAT,BUFFER)
1420 CALL DETACH(01,ISTAT,BUFFER)
1430 STOP
1440       END
```

PROGRAM NET1

SUBROUTINE PRINT(MODE)

Purpose: To print node designations and text for all nodes stored in array LLX. MODG=0; Print Foreward; Mode=1, Print Back; Mode 2; print rejected nodes.

```
1450       SUBROUTINE PRINT(MODE)
1460       DIMENSION GTO(600),LIN(600),LOC(600),NBR(600),NOD(600),NOF(60
1470      &,NEV(8),NST(8),LLX(8,90)
1480      &,MES(9,1500)
1490       COMMON GTO,LIN,LOC,NBR,NOD,NOF,MES,NST,LLX,NEV
1500      &,WORDS,NM1,NM2,NM3,MAXLN,IFO
1510       DIMENSION ISET(90),IOUT(8,90),JOUT(90,8),KOUT(90,8)
1520       INTEGER WORDS,GTO
1530       DATA NO,LANK/2HNO,4H      /
```

This section prints the node designations (without the first 7 characters)

of the ncdes in

of the tree being printed, but lines 1550-1600 may be removed without affecting the major functions of the program.

```
1540          MZ=0
1550          WRITE(IFO,38) MODE
1560          DO 39  K=1,8
1570          LE=NEV(K)
1580          WRITE(IFO,38) (LLX(K,J),J=1,LE)
1590      39  CONTINUE
1600      38  FORMAT(11I6)
```

The nodes having

to 0.

```
1540          MZ=0
1610          DO 50  I=1,50
1620      50  ISET(I)=0
1630          IS=0
```

The node designations stored in LLX are converted to node indices

```
1640      70  DO 84  I=1,8
1650          LIM=NEV(I)
1660          IF (LIM)  84,84,92
1670      92  IF (I-3)  94,94,82
1680      94  LLX(I,1)=I
1690          IOUT(I,1)=I
1700          GO TO 84
```

SUBROUTINE PRINT (continued)

```
82 DO 83  K=1,LIM
   CALL FIND(LLX(I,K),NOD,KK)
   IF (KK.NE.999) GO TO 78
   PRINT 79,LLX(I,K)
79 FORMAT(6H NODE  ,I5,10H NOT FOUND )
   GO TO 83
```

(NOTE: The nodes in LLX were computed from the initial node and the number of branches in each parent node. If there is an error in the number of branches or if the node has been ommitted from the data file, this error message may be printed. Program execution will continue. If the error lies in specifying a value for the number of branches larger than the correct value, the output will be correct. It will, of course, be in error if a node has been omitted.)

```
78 LLX(I,K)=KK
   IF (GTO(KK))  83,83,81
81 IS=IS+1
   ISET(IS)=KK
   MZ=IS
83 CONTINUE
84 CONTINUE
```

The node indices are transferred to array 'IOUT'. This array is similar to 'LLX' except that blank entries are inserted to provide provide spacing of the output for levels 4 - 8.

```
   PRINT,MZ
   DO 90  LM=4,8
   MAX=NEV(LM)
   IF (MAX)  90,90,91
91 IT=0
   DO 86  I=1,MAX
   IZZ=LLX(LM,I)
   IF (IZZ)  86,86,80
80 IT=IT+1
   IOUT(LM,IT)=IZZ
   IF (MODE)  10,10,86
10 NBB=NBR(IZZ)
   IF (NBB-1)  86,86,72
72 DO 74  KK=2,NBB
   IT=IT+1
74 IOUT(LM,IT)=0
86 CONTINUE
   NEV(LM)=IT
90 CONTINUE
```

The tree is printed in sections, two levels per section. The major local variables in this section are:

LVL - Index of the lower level (printed at the left of the page.

LVH - Index of the upper level (printed at the right of the page)

LO - Number of nodes in the lower level

LH - Number of nodes in the upper level

L1 - Larger of the two, LO and LH

NLO - Index of the higher level node to be printed

NHI - Index of the upper level node to be printed

```
      DO 130  KZ=1,8,2
      LVL=8-KZ
      LVH=9-KZ
      LH=NEV(LVH)
      LO=NEV(LVL)
```

NOTE: If LO=0 and LH=0 and this was a print forward, the printing is finished. Otherwise, the section heading is printed.

If the lower level is 1, the node designations are printed at this point.

```
      IF (LO+LH)  95,95,101
 95   IF (MODE)  223,223,130
 101  WRITE(IFQ,111)LVL
 111  FORMAT(//28X,15HNETWORK SECTION   ,I2/)
      IF (LVL-1)  125,125,128
 125  WRITE(IFQ,901)NM1,NM1,NM2
 901  FORMAT(////1X,A3,36X,A3,A2)
```

The output arrays, JOUT (left side) and KOUT (right side), are set to blanks.

```
      NLO=1
      NHI=2
 128  L7=0
      L8=0
      L1=LH
      IF (LO.GT.LH)  L1=LO
      DO 129  K=1,L1
      DO 103  LN=1,MAXLN
      DO 103  LW=1,WORDS
      KOUT(LN,LW)=LANK
      JOUT(LN,LW)=LANK
 103  CONTINUE
```

PROGRAM NET1

SUBROUTINE PRINT (continued)

The local variables, NLO and NHI, are set to the indices of the lower and upper level nodes next to be printed. If there is no higher level node, then the local variables NFF and NDD are set to blank and 0, respectively. If there is a higher level node, NDD is set to the node designation and NFF to the final alpha characters in the full node designation (blank except for level 8 nodes). If the higher level is level 8, the final integer added to the node designation during the input phase must be removed before printing.

```
      IF (LVL.EQ.1)  GO TO 108
      NLO=IOUT(LVL,K)
      NHI=IOUT(LVH,K)
      LA=NLO + NHI
      LB=K-L1
      IF(LA.EQ.0.AND.LB.EQ.0) GO TO 130
   29 IF (NHI)  40,40,42
   40 NFF=LANK
      NDD=0
      GO TO 46
   42 NFF=NOF(NHI)
      NDD=NOD(NHI)
   44 IF (LVH.EQ.8) NDD=NDD/10
```

Full node designations are printed.

```
   46 IF (LH)  96,96,98
   96 WRITE(IFQ,903)NM1,NM2,NM3,NOD(NLO)
  903 FORMAT(////1X,A3,2A2,I4)
      L8=NHI
      GO TO 108
   98 IF (NLO)  106,106,104
  104 WRITE(IFQ,102)NM1,NM2,NM3,NOD(NLO),NOF(NLO),NM1,NM2,NM3,
     &NDD,NFF
  102 FORMAT(////1X,A3,2A2,I4,A4,24X,A3,2A2,I4,A4)
      GO TO 108
  106 WRITE(IFQ,902)NM1,NM2,NM3,NDD,NFF
  902 FORMAT(////41X,A3,2A2,I5,A4)
```

[illegible] for the nodes are transferred to the output arrays, JOUT and KOUT, [illegible] printed.

PROGRAM NET1

SUBROUTINE PRINT (continued)

```
  108 IF (NLO-L7) 114,116,ii4
  114 LS7=LOC(NLO)
      LE7=LS7+LIN(NLO)-1
      LP=0
      DO 115  JM=LS7,LE7
      LP=LP+1
      DO 115  JW=1,WORDS
  115 JOUT(LP,JW)=MES(JW,JM)
      L7=NLO
  116 IF (NHI.EQ.0) GO TO 120
      IF (NHI-L8)  118,120,118
  118 LS8=LOC(NHI)
      LE8=LS8+LIN(NHI)-1
      LP=0
      DO 119  JM=LS8,LE8
      LP=LP+1
      DO 119  JW=1,WORDS
  119 KOUT(LP,JW)=MES(JW,JM)
      L8=NHI
  120 DO 122 LZ=1,MAXLN
  122 WRITE(IFQ,117)(JOUT(LZ,LX) ,LX=1,WORDS),(KOUT(LZ,LY),LY=1,WOR
  117 FORMAT(1X,8A4,4X,8A4)
  129 CONTINUE
  130 CONTINUE
  223 DO 200  I=1,WORDS
      DO 200  J=1,50
      LLX(I,J)=0
      IOUT(I,J)=0
  200 CONTINUE
```

If any of the nodes printed had GO TO lines, the user would be given the option of having them printed.

```
      IF (MZ)  999,999,899
  899 PRINT 905
  905 FORMAT(16H LOOK AT GO TO'S  )
      READ 906,IWS
  906 FORMAT(A2)
      IF (IWS.EQ.NO)  GO TO 999
      WRITE(IFQ,907)
  907 FORMAT(5H NODE  ,10X,5HGO TO  )
      DO 910  I=1,MZ
      J=ISET(I)
      LS=LOC(J)
      LE=LS+LIN(J)-1
```

PROGRAM NET1

SUBROUTINE PRINT (continued)

```
      DO 916  K=LS,LE
      IF (MES(9,K))  916,916,912
  912 WRITE(IFO,914)NM1,NM2,NM3,NOD(J),(MES(L,K),L=1,8)
  914 FORMAT(1X,A3,2A2,I4,1X,8A4)
  916 CONTINUE
  910 CONTINUE
```

Control is returned to the MAIN CONTROL PROGRAM.

```
  999 RETURN
      END
```

SUBROUTINE SF(LLX,NEV,NOD,NBR,NOD,LS)

Purpose: To identify and store in LLX all higher level nodes directly connected to the initial node, NODE.

LS is the level of NODE and array NEV will contain the number of nodes to be printed in each level.

```
 2980       SUBROUTINE SF(LLX,NEV,NOD,NBR,NODE,LS)
 2990       DIMENSION LLX(8,50),NEV(8),NOD(600),NBR(600)
```

The node counter, NEV, is set to 0.

```
 3000       DO 8  I=1,8
 3010     8 NEV(I)=0
```

Program control is transferred to instructions for level LS.

```
 3020       GO TO (10,10,20,30,30,30,70,80),LS
```

Levels 1 and 2. (NOTE: Currently, the initial node may not be in levels 1 or 2)

```
 3030    10 PRINT,"ERROR IN INITIAL NODE LEVEL"
 3040       LS=99
 3050       GO TO 90
```

Levels 3 - 6

```
3060    20 LLX(3,1)=0
3070       NEV(3)=1
3080       GO TO 31
3090    30 LLX(LS,1)=NODE
3100       NEV(LS)=1
3110    31 DO 32  I=LS,6
3120       J=I
3130       K=I+1
3140       CALL CHILD(J,K,LLX,NOD,NBR,NEV)
3150    32 CONTINUE
3160       GO TO 75
```

PROGRAM NET1

SUBROUTINE SF (continued)

Level 7

```
   70 LLX(7,1)=NODE
      NEV(7)=1
   75 NG=NEV(7)
      NEV(8)=NG
      DO 77  K=1,NG
   77 LLX(8,K)=LLX(7,K)*10+1
      GO TO 90
```

Level 8

```
   80 LLX(8,1)=NODE*10+1
      NEV(8)=1
```

Control is returned to the MAIN CONTROL PROGRAM

```
   90 RETURN
      END
```

PROGRAM NET1

SUBROUTINE SB(LLX,NOD,NEV,LS,NODE)

Purpose: To identify and store in LLX all lower level nodes directly connected to the initial node, NODE.

LS is the level of MODE and array NEV will contain the number of nodes to be printed in each level.

```
  3280        SUBROUTINE SB(LLX,NOD,NEV,LS,NODE)
  3290        DIMENSION LLX(8,50),NOD(600),NEV(8)
```

The node counter, NEV, is set to 0. The initial node, NODE, is stored in LLX.

```
3300        DO 12  I=1,8
3310     12 NEV(I)=0
3320      2 LLX(LS,1)=NODE
3330        NEV(LS)=1
```

The next lower level is computed. If it is above 3, the "parent" node in that level is stored.

```
3340        LS=LS-1
3350        IF (LS-4)  20,10,10
3360     10 NODE=NODE/10
3370        GO TO 2
```

If the level is 3, then the nodes in levels

```
3380     20 DO 22  K=1,3
3390        NEV(K)=1
3400        LLX(K,1)=NOD(K)
3410     22 CONTINUE
```

Control is returned to the MAIN CONTROL PROGRAM.

```
3420        RETURN
3430        END
```

PROGRAM NET1

SUBROUTINE SR(LLX,NOD,NEV,LS,NODE,NBR)

Purpose: To identify and store in LLX all lower levels not directly connected with initial node, NODE, but in the same main branch of the tree.

```
3440        SUBROUTINE SR(LLX,NOD,NEV,LS,NODE,NBR)
3450        DIMENSION LLX(8,50),NOD(600),NEV(8)
```

The node counter, NEV, is set to 0. The initial node, NODE, is stored in LLX.

```
3460       &,NBR(600)
3470        DO 6  I=1,8
3480      6 NEV(I)=0
```

The next lower level is computed. If it is above 3, the Parent and Grandparent Nodes are computed.

```
3490        NEV(LS)=1
3500        LLX(LS,1)=NODE
3510      9 LP=LS-1
3520        IF (LP-3)  20,20,10
3530     10 NODP=NODE/10
3540        NODG=NODP/10
```

If the grandparent Node is 0, its level must be 3; otherwise, subroutine FIND is called to get level K of the grandparent NODG.

```
3550        IF (NODG)  30,30,32
3560     30 K=3
3570        GO TO 34
3580     32 CALL FIND(NODG,NOD,K)
```

All children of the grandparent Node are stored in LLX.

```
3590     34 MX=NBR(K)
3600        DO 12  J=1,MX
3610     12 LLX(LP,J)=NODG*10+J
3620        NEV(LP)=MX
```

PROGRAM NET1

SUBROUTINE SR (continued)

The next lower level is considered and the steps above repeated until level 3 is reached.

```
3630        LS=LP
3640        NODE=NODP
3650        GO TO 9
```

When level 3 is reached, no further searching is necessary. Levels 1, 2, and 3 have only one node each. These nodes are stored in LLX.

```
3660    20 DO 22  J=1,3
3670        LLX(J,1)=NOD(J)
3680        NEV(J)=1
3690    22 CONTINUE
```

Control is returned to the MAIN CONTROL PROGRAM.

```
3700        RETURN
3710        END
```

PROGRAM NET1

SUBROUTINE CHILD(L1,L2,LLX,NOD,NBR,NEV)

Purpose: To store in LLX all children of those level L1 nodes already in LLX.

```
      SUBROUTINE CHILD(L1,L2,LLX,NOD,NBR,NEV)
      DIMENSION LLX(8,50),NOD(600),NBR(600),NEV(8)
```

The local variable, LIM, is set to the number of L1 nodes in LLX. N will be the counter for the level L2, nodes to be stored.

```
      LIM=NEV(L1)
      N=0
```

The index of each node stored in LLX must be found.

```
      DO 30  I=1,LIM
      K=LLX(L1,I)
      IF (K)  11,11,12
   11 L=3
      GO TO 26
   12 CALL FIND(K,NOD,L)
```

(NOTE: If the node cannot be found, an error message will be printed at the terminal. Execution will continue, but the output for this option will not be correct.)

```
      IF (L.NE.999) GO TO 26
      PRINT 100,L1,K
  100 FORMAT(25HERROR IN SUBROUTINE CHILD/5HNODE ,I6,5HLEVEL,1X,I3,
     &39HCANNOT BE FOUND: EXECUTION CONTINUING.  )
      GO TO 30
```

All children of this node are stored in LLX. These steps are repeated for all level L1 nodes in LLX. When all nodes have been processed, control is returned to subroutine SF.

```
   26 KK=K*10
      MIM=NBR(L)
      DO 27  NQ=1,MIM
      N=N+1
   27 LLX(L2,N)=KK+NQ
      NEV(L2)=NEV(L2)+MIM
   30 CONTINUE
      RETURN
      END
```

PROGRAM NET1

MINOR UTILITY SUBROUTINES

SUBROUTINE FIND(I,N,K)

Purpose: To find the index K of the node designated by I.

```
      SUBROUTINE FIND(I,N,K)
      DIMENSION N(600)
      DO 10  K=1,600
      IF (I.EQ.N(K)) GO TO 12
   10 CONTINUE
      K=999
   12 RETURN
      END
```

SUBROUTINE LEVEL(LL,I,LS)

Purpose: To find the level LS of node I.

```
      SUBROUTINE LEVEL(LL,I,LS)
      DIMENSION LL(7)
```

Array LL contains powers of 10 (set in the MAIN PROGRAM). The search for LS is based on the numerical value of the node designation.

```
      DO 10  J=1,5
      LS=6-J
      IF (I-LL(LS))  10,12,12
   10 CONTINUE
      LS=LS-1
   12 LS=LS+3
      RETURN
      END
```

Appendix II: Annotated Program Listings for NET2

PROGRAM NET2

Purpose: To identify and print all groups of equivalent nodes across the entire set of trees.

```
 10       DIMENSION KON(5),N(5),BUFFER (380),
 20      &NN(5010,5),II(5010),LR(5010),LA(5010),IG(5010)
 30        INTEGER BL,AS
 40       DATA BL,AS/1H ,1H*/
 50       DATA KON/1HE,1HM,1HP,1HS,1HT/
```

The input file NIF (Node Information File) is attached.

```
60        I=0
70        CALL ATTACH (01, "AWCISS/CACI",1,0,ISTAT,BUFFER)
```

Input phase. The input file is read. The local variables I1, I2, and I3 (containing actor, action, and target codes, respectively) are packed into a single word. The other node descriptors are stored.

```
80     1 READ (1,100,END=50) N,I1,I2,I3,I4,I5
90   100 FORMAT(3A1,2A4,1X,I3,3X,I2,1X,I3,3X,I1,I2)
100        IF (N(1).EQ.AS) GO TO 50
110        K=I1*100000+I2*1000+I3
120  I=I+1
130        IF (I.GT.5010) GO TO 50
140        II(I)=K
150        LR(I)=I4
160        LA(I)=I5
170        DO 8  M=1,5
180      8 NN(I,M)=N(M)
```

The equivalence group for the node is determined. Either it is equivalent to a previous node, or its descriptors form the basis of a new group.

```
      IF (I-1)  10,10,20
   10 IG(1)=1
      LG=1
      GO TO 1
   20 KFND=I-1
      DO 28  M=1,KEND
      IF (II(M)-K)  28,24,28
   24 IF (LA(M).EQ.LA(I).AND.LR(M).EQ.LR(I)) GO TO 30
   28 CONTINUE
      LG=LG+1
      IG(I)=LG
      GO TO 1
   30 IG(I)=IG(M)
      GO TO 1
```

The input file has been read; the number of records and groups are printed.

```
   50 NR=I
      PRINT 101,NR,LG
  101 FORMAT(" INPUT COMPLETE ",I5," RECORDS  ",I5,"  GROUPS")
```

For each tree type; E,M,P,S, and T

```
      DO 60  I=1,5
      WRITE (9,102) KON(I)
  102 FORMAT(1H1," EQUIVALENT NODES FOR  ",A1," TRFS"/)
      DO 58  J=1,NR
      DO 52  IJ=1,3
      IF (NN(J,IJ).NE.BL)  GO TO 53
   52 CONTINUE
      GO TO 58
   53 IF (NN(J,IJ).NE.KON(I))  GO TO 58
      LGX=IG(J)
```

Print each node belonging to it ···

```
      WRITE(9,103) (NN(J,KA),KA=1,5),II(J),LR(J),LA(J)
  103 FORMAT(1H ,4HNODE,1X,3A1,2A4,4X,16HFOUIVALENT NODES
     &3X,I10,2(1X,I3))
```

...and each member in its equivalence group.

PROGRAM NET2 (continued)

```
490        DO 56  JJ=1,NR
500        IF (IG(JJ).NE.LGX)  GO TO 56
510        IF (J-JJ)  54,56,54
520     54 WRITE (9,104) (NN(JJ,KA),KA=1,5),II(JJ),LR(JJ),LA(JJ)
530    104 FORMAT(24X,3A1,2A4,5X,I10,2(1X,I3))
540     56 CONTINUE
550     58 CONTINUE
560        WRITE(9,105) KON(I)
570    105 FORMAT(2X,"END OF EQUIVALENT NODES FOR ",A1," TREE"/)
580     60 CONTINUE
```

The printing is finished.

```
604        PRINT 605
620    605 FORMAT(" RUN FINISHED")
```

PROGRAM execution is ended.

```
610        STOP
620        END
```

Appendix III: Annotated Program Listing for NET3

PROGRAM NET3

Purpose: To select and print nodes from the Node Information File (NIF).

MAIN CONTROL PROGRAM

Purpose: To identify user requests and to read the file and print selected nodes.

```
-----------------------------------------------------------------------------
10        DIMENSION IWORD(5),IN(5),N(3),BUFFER(380)
20       &,KN(5),NS(3)
30        INTEGER BL,ALL
40        DATA BL,NO,IO,IH,ALL/1H ,1HN,1HO,1HH,1HA/
50        DATA IWORD/3HATR,3HACT,3HTAR,3HREG,3HSUB/
-----------------------------------------------------------------------------
```

The input file, NIF *is attached. Statement* 100 *is the format by which the* file will be read.

```
-----------------------------------------------------------------------------
 65        CALL ATTACH(01,"AWC1SS/CACI/NIF;",1,0,ISTAT,BUFFER)
 80   100 FORMAT(3A4,I3,3X,I2,1X,I3,3X,I1,I2)
 190  101 FORMAT(1X,14HNO. RECORDS = ,I6)
-----------------------------------------------------------------------------
```

The next section solicits and reads the user's instructions. His selection options are stored in IWORD: he may select nodes with a specific actor, action, target, region, or substantive area. As each option is presented, he enters either a specific code or the word ALL, indicating that he does not wish to select nodes by this variable. A request for help at the beginning will result in printing his options (see format statement 105, line 310). Entering the word HELP in response to requests for codes will cause the program to print the name of the variable for which a code is requested. The word OUT entered at any point will cause the HELP option to be deactivated.

```
-----------------------------------------------------------------------------
200    18 PRINT 102
210   102 FORMAT(1X,11HWANT HELP  )
220    20 READ 103,INS
230   103 FORMAT(3A1)
240       IF (INS.EO.NO)  GO TO 22
250       IF (INS.NE.BL) GO TO 21
260       PRINT 104
270   104 FORMAT(1X,69HMISSED YOUR ANSWER. PLEASE RE-ENTER IT. ST
280      &HE FIRST COLUMN.  )                              ARTING IN T
290       GO TO 20
```

PROGRAM NET3 (continued)

Note:

```
 21 PRINT 105
105 FORMAT(70H YOU MAY SELECT NODES BY ONE OR MORE OF THE FOLLOW,
      &DE ATTRIBUTES:  /11H ACTOR(ATR) /9H ACT(ACT) / 12H TARGET(TA
      &22H SUBSTANTIVE AREA(SUB)  / 12H REGION(REG) /        ING NO
      &43H EACH ATTRIBUTE WILL BE CONSIDERED IN TURN   /   R) /
      &32H ENTER A SPECIFIC CODE OR 'ALL'  /
      &" ENTER  CODES AS 3-DIGIT INTEGERS; I.E. ACTOR '2' AS '002'/
       & REGION '3' AS '003'"/
      &" TO REQUEST HELP AT ANY STAGE, ENTER 'HELP'"/
      &28H TO CANCEL HELP, ENTER 'OUT'  )
 22 DO 40  J=1,5
    IF (INS.EQ.NO) GO TO 25
 24 PRINT 106,IWORD(J)
106 FORMAT(12H SELECT ONE  ,A3)
 25 READ 103,NS
    IF (NS(1).EQ.ALL) GO TO 30
    IF (NS(1).EQ. IO) GO TO 28
    IF (NS(1).EQ. IH) GO TO 27
 26 CALL CONV(NS,KI)
    IF (KI.EQ.9999) GO TO 24
    IN(J)=KI
    GO TO 40
 27 INS=BL
    GO TO 24
 28 INS=NO
    GO TO 25
 30 IN(J)=-1
 40 CONTINUE
```

The user-selected codes are stored in the array, IN. When the user input is complete, this array is printed as a check. A value of -1 for any word in this array indicates that the corresponding variable will not be used in selecting nodes to be printed (for example, all values of this variable will be accepted).

```
      PRINT 200,IN
200  FORMAT(5I4)
```

The counter, ITOT, is set to 0 and the input file is rewound. As each file record is read, it is compared with the set of selection codes in IN; all matches are printed.

```
      ITOT=0
      REWIND 1
    1 READ(1,100,END=50) N,KN
```

PROGRAM NET3 (continued)

```
610         DO 48  J=1,5
620         IF (IN(J)) 48,42 ,42
630     42  IF (KN(J).NE.IN(J)) GO TO 1
640     48  CONTINUE
650         ITOT=ITOT+1
660         PRINT 100,N,KN
670         GO TO 1
```

When the reading of the file is finished, the number of matches is printed and the user is given a chance to make another search through the file. If his answer is NO, the input file is detached and program execution is terminated.

```
690    50 PRINT 108,ITOT
700   108 FORMAT(/I6,1X,7HMATCHES// 21H ADDITIONAL SEARCHES      )
710       READ 103,INS
720       IF (INS.NE.NO) GO TO 18
725       CALL DETACH(01,ISTAT,BUFFER)
730       STOP
740       END
```

PROGRAM NET3

SUBROUTINE CONV(N,KI)

Purpose: To convert the 3-character alpha variable, N, into a 3-digit integer, KI.

```
      SUBROUTINE CONV(N,KI)
       DIMENSION NUM(10),N(3),L(3)
      DATA NUM/1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9,1H0/
      DATA L/100,10,1/
```

Each character in N is compared with the list of integers in array NUM to find a match. If there is an error, a message is printed.

```
      KI=0
      DO 20  K=1,3
      DO 10  J=1,10
      IF (N(K).EQ.NUM(J)) GO TO 12
   10 CONTINUE
      PRINT 100,N
  100 FORMAT(1X,9HERROR IN ,3A1)
      KI=9999
      GO TO 99
```

The 3-digit integer is constructed from powers of 10 and the match J.

```
   12 IF (J.EQ.10)  J=0
      KI=KI+J*L(K)
   20 CONTINUE
```

Control is transferred back to the MAIN CONTROL PROGRAM.

```
   99 RETURN
      END
```

Appendix IV: Annotated Program Listing for NET4

PROGRAM NET4

Purpose: To modify and update the Tree Summary File (TSF); to retrieve information from the file for specific trees.

MAIN PROGRAM

Purpose: To read the TSF and to execute user instructions.

```
      DIMENSION II(29),JCODE(7)
      DIMENSION IX(60,29),IS(7),KS(7),BUFFER(380)
 DATA IS/3HHEL,3HALL,3HMOD,3HADD,3HDIS,3HEND,3HSTO/
      DATA KS/3HAUT,3HDTC,3HDTM,3HACT,3HSUB,3HREG,3HTRE/
      DATA NO,LANK/2HNO,4H      /
      DATA NC,NS,NN,NV/7,7,60,29/
```

The TSF is located and read. (It is stored in the AWC system under the name DNET4.)

```
      CALL ATTACH(01,"AWCISS/CACI/DNET4;",1,0,ISTAT,BUFFER)
      NUM=0
      I=0
    1 I=I+1
      IF (I.GT.NN) GO TO 4
       READ(1,100,END=4) (IX(I,J),J=1,NV)
  100 FORMAT(2A4,1X,A4,1X,2(I6,1X),I3,1X,A3,2(1X,A2)/20A4)
       IF (IX(I,1).NE.LANK) GO TO 1
    4 NR=I-1
       PRINT 102,NR
  102 FORMAT(16H INPUT COMPLETE ,I4,14H RECORDS READ   )
```

The user's option is read and control transferred to the appropriate program instruction.

```
    6 PRINT 104
  104 FORMAT(13H ENTER OPTION )
      READ 105,INS
  105 FORMAT(A3)
      DO 8  I=1,NC
      IF (INS.EQ.IS(I)) GO TO 9
    8 CONTINUE
      GO TO 10
    9 GO TO (10,20,30,40,50,60,70),I
```

PROGRAM NET4 (continued)

The user has requested help; the program options are printed.

```
250    10 PRINT 106
260   106 FORMAT(38H THE FOLLOWING OPTIONS ARE AVAILABLE:  /
270      &34H HELP - TO PRINT USER INSTRUCTIONS /
280      &28H ALL - TO PRINT ENTIRE FILE /
290      &33H MOD - TO MODIFY LAST CHANGE DATE  /
300      &29H ADD - TO ADD ENTRIES TO FILE  /
310      &40H DIS - TO DISPLAY SELECTED FILE ENTRIES  /
320      &33H STOP - TO STOP PROGRAM EXECUTION )
330       GO TO 6
```

The user has requested that the entire file be printed.

```
340    20 DO 22  J=1,NR
350    22 PRINT 100,(IX(J,K),K=1,NV)
360   108 FORMAT(I3,1X,A4,1X,2A4,1X,2(I6,1X),I3,1X,A3,2(1X,A2)/20A4)
370       GO TO 6
```

The last modification date of a specific tree is to be updated; The user indicates the tree name and the new date.

```
380    30 PRINT 110
390   110 FORMAT(12H TREE NAME   )
400       READ 100,IA,IB
402       IF (IA.NE.IS(1)) GO TO 31
404       PRINT,"TREE NAMES CONSIST OF 7 CHARACTERS; FOR EXAMPLE: T010101"
406       GO TO 6
410    31 DO 32  J=1,NR
420       IF (IA.EQ.IX(J,1).AND.IB.EQ.IX(J,2)) GO TO 34
430    32 CONTINUE
440       PRINT 114,IA,IB
450   114 FORMAT(10H ERROR IN ,2A4)
460       GO TO 30
470    34 PRINT 115
480   115 FORMAT(22H DATE OF LAST CHANGE )
490       READ 116,IDT
500   116 FORMAT(I6)
```

The user's instructions are rechecked. Upon his OK, the new date is written into the file.

```
510       PRINT 118,IX(J,4),IDT
520   118 FORMAT(10H OLD DATE ,I6,19H TO BE REPLACED BY ,I6)
```

PROGRAM NET4 (continued)

ACT- to select entries by actor

AUT- to select entries by author

DFC- to select file entries by file creation date

DFM- to select entries by the date of last modification

SUB- to select entries by substantive area

REG- to select entries by region

```
800    50 DO 51   K=1,NS
810    51 JCODE(K)=LANK
820    52 PRINT 130
830   130 FORMAT(23H ENTER SELECTION OPTION   )
840       READ 105,INS
850       IF (INS.EQ.NO) GO TO 57
860       IF (INS.EQ.IS(1)) GO TO 59
870       IF (INS.EQ.IS(6)) GO TO 6
880       IF (INS.EQ.IS(7)) GO TO 70
890       DO 53  J=1,NS
900       IF (INS.EQ.KS(J)) GO TO 54
910    53 CONTINUE
920       PRINT 114,INS
930       GO TO 59
940    54 IF (J.NE.NS)  GO TO 55
```

The user wishes to print information on a specific tree. He is asked to enter the name of the tree. If the tree is in the file, the entry is printed.

```
950       PRINT 110
960       READ 100,IA,IB
970       DO 502  J=1,NR
980       IF (IA.NE.IX(J,1)) GO TO 502
990       IF (IB.NE.IX(J,2)) GO TO 502
1000       PRINT 100,(IX(J,K),K=1,NV)
1010       GO TO 50
1020   502 CONTINUE
1030       PRINT 114,IA,IB
1040       GO TO 52
```

The user has indicated one of the specific entry attributes: author, date, actor, substantive area, or region. Program control is transferred to the proper instruction to read the specific code for the attribute selected. The code entered by the user is stored in the selection code array,
may end this section by entering the word NO.

```
1050    55 GO TO (510,520,520,540,550,560),J
```

PROGRAM NET4 (continued)

```
530       PRINT 120
540   120 FORMAT(5H OK )
550       READ 122,INS
560   122 FORMAT(A2)
570       IF (INS.EQ.NO) GO TO 30
580       IX(J,4)=IDT
590       NUM=NUM+1
600       GO TO 6
```

The user wishes to add an entry to the file. If the file is full, he is instructed to increase the array dimensions before adding new entries. After his new entry has been typed, he is given the opportunity of checking it against the last entry in the old file. If he is satisfied that the lines are correct, they are added to the file.

```
610    40 IF (NR-NN) 41,48,48
620    41 PRINT 124
630   124 FORMAT(15H ENTER NEW LINE )
640       READ 100,II
650       PRINT 126
660   126 FORMAT(20H LAST TWO LINES ARE:   )
670       PRINT 100,(IX(NR,K),K=1,NV)
680       PRINT 100,II
690       PRINT 120
700       READ 122,INS
710       IF (INS.EQ.NO)  GO TO 40
720       NR=NR+1
730       DO 42  J=1,NV
740    42 IX(NR,J)=II(J)
750       NUM=NUM+1
760       GO TO 40
770    48 PRINT 128
780   128 FORMAT(40H FILE FULL. DIMENSIONS MUST BE INCREASED   )
790       GO TO 6
```

The user wishes to display selected portions of the file. He is instructed to enter his selection option. His choices at this point are:

NO- to indicate that he has completed the selection code array
and wishes the matching entries to be printed,

HELP- to print the list of options

STOP- to terminate program execution

TREE- to print the entry for a specific tree

PROGRAM NET4 (conitnued)

```
1060   510 PRINT 134
1070   134 FORMAT(13H ENTER AUTHOR )
1080       READ 112,JCODE(1)
1085   112 FORMAT(A4)
1090       GO TO 52
1100   520 PRINT 136
1110   136 FORMAT(20H ENTER EARLIEST DATE )
1120       READ 116,JCODE(J)
1130       GO TO 52
1140   540 PRINT 137
1150   137 FORMAT(12H ENTER ACTOR   )
1160 READ 105,JCODE(4)
1162 GO TO 52
1170   550 PRINT 138
1180   138 FORMAT(23H ENTER SUBSTANTIVE AREA   )
1190 READ 122,JCODE(5)
1192 GO TO 52
1200   560 PRINT 140
1210   140 FORMAT(13H ENTER REGION )
1220 READ 122,JCODE(6)
1230       GO TO 52
```

All selection options have been read. The file is searched and all entries matching the selected codes are printed.

```
1240    57 ITOT=0
1250       DO 574  J=1,NR
1260       IF (JCODE(1).EQ.LANK) GO TO 571
1264       IF (IX(J,3).NE.JCODE(1)) GO TO 574
1266   571 IF (JCODE(2).EQ.LANK) GO TO 572
1270       IF (IX(J,4).LT.JCODE(2)) GO TO 574
1272   572 IF (JCODE(3).EQ.LANK) GO TO 575
1274       IF (IX(J,5).LT.JCODE(3)) GO TO 574
1276   575 IF (JCODE(4).EQ.LANK) GO TO 576
1278       IF (IX(J,7).NE.JCODE(4)) GO TO 574
1280   576 IF (JCODE(5).EQ.LANK) GO TO 577
1282       IF (IX(J,8).NE.JCODE(5)) GO TO 574
1286   577 IF (JCODE(6).EQ.LANK) GO TO 578
1288       IF (IX(J,9).NE.JCODE(6)) GO TO 574
1290   578 ITOT=ITOT+1
1300       PRINT 100,(IX(J,L),L=1,NV)
1310   574 CONTINUE
1320       PRINT 142,ITOT
1330   142 FORMAT(1X,I3,8H MATCHES )
1340       GO TO 50
```

PROGRAM NET4 (conitnued)

The user has requested help; the selection options are printed.

```
1350    59 PRINT 144
1360   144 FORMAT(" SELECTION OPTIONS ARE:" / " TREE - TREE NAME " /
1370       &15H AUTH - AUTHOR  / 20H DTC - DATE CREATED  /
1380       &" DTM - DATE MODIFIED" / "A ACT - ACTOR" /
1390       &15H SUB - SUB-AREA  / 13H REG - REGION  /
1400       &28H END - TO END CURRENT OPTION  )
1410        GO TO 50
```

The user has entered the instruction to END. To avoid confusion over whether he meant to end the current option or to end the program, he is given an opportunity to recheck his instruction. Upon his OK, program control is transferred to the ending sequence.

```
1420    60 PRINT 146
1430   146 FORMAT(10H END NOW )
1440        READ 105,INS
1450        IF (INS.NE.NO)  GO TO 6
```

If the file has been modified, the user is asked to indicate whether or not he wishes the modified file to be saved.

```
1460    70 IF (NUM)  80,80,72
1470    72 PRINT 148,NUM
1480   148 FORMAT(I3,38H MODIFICATION(S). SAVE MODIFIED FILE    )
1490       READ 105,INS
1500       IF (INS.EQ.NO) GO TO 80
1510       REWIND 1
1518       DO 78  I=1,NR
1520       WRITE(1,100) (IX(I,J),J=1,NV)
1522    78 CONTINUE
1530       PRINT 150
1540   150 FORMAT(11H FILE SAVED )
```

The input file is released and program execution is terminated.

```
1550    80 CALL DETACH(01,ISTAT,BUFFER)
1560       STOP
1570       END
```

Appendix V: Listing of FORECAST 90 Files

```
2574T 01  08-05-75    12.878              F I L E - S Y S T E M   A C T I V I T Y                              PAGE      1

                     2   CLIST AWCISS/CACI                                                                  PRIOR STATUS ON PG

    CATALOG NAME     SCC          GEN'L DEVICE    CREATE        DATE OF LAST       TIME      MAX CURRENT LAST SAVE ALLOC  FILE
      FILE NAME                   PERMS TYP:NAME   DATE   MODIFY ALLOC  CHANGE            LLINKS LLINKS VOL-CHG?  CNT  STAT TYPE

CACI                                    190:ST1   062075 062075
      NET1           UNC                190:DP2   062075 062675 080575 071875 09.593     50      13  09323N     154        SEQ
            PERMISSIONS: R/WAWC-DS/, W/AWC-DS/
      NET2           UNC                190:DP2   062075 062075 071275 070875 10.502     50      13  09323N      11        SEQ
      NET3           UNC                190:DP2   062075 062075 080175 070375 12.453     50      13  09323N      14        SEQ
      NET4           UNC                190:DP2   062075 062075 080575 070175 20.008     50      13  09323N      18        SEQ
      DNET4          UNC                190:ST1   062075 062075 080575 071675 10.475     50      13  09323N      18        SEQ
      E010201        UNC                190:DP2   071275 071275 071875 071575 18.386    520      26  09323N       4        SEQ
      E010104        UNC                190:DP2   071275 071275 071575 071575 18.386    520      26  09323N       2        SEQ
      S010401        UNC                190:DP2   071275 071275 071575 071575 18.386    520      26  09323N       2        SEQ
      P010501        UNC                190:DP2   071275 071275 071575 071575 18.511    520      26  09323N       3        SEQ
      S010501        UNC                190:DP2   071275 071275 071575 071575 18.511    520      26  09323N       2        SEQ
      E010102        UNC                190:DP2   071275 071275 071675 071675 09.348    520      26  09323N       4        SEQ
      E010301        UNC                190:DP2   071275 071275 071675 071575 17.333    520      26  09323N       7        SEQ
      E010302        UNC                190:ST1   071275 071275 071575 071575 18.512    520      26  09323N       4        SEQ
      T010201        UNC                190:DP2   071275 071275 071575 071575 15.567    600      30  09323N       8        SEQ
      M010401        UNC                190:ST1   071275 071275 071575 071275 15.651    520      26  09323N       3        SEQ
      E010101        UNC                190:ST1   071275 071275 080575 071575 18.334    520      26  09323N       4        SEQ
      P010802        UNC                190:DP2   071275 071275 071575 071575 18.334    520      26  09323N       2        SEQ
      E010103        UNC                190:DP2   071275 071275 071575 071575 18.334    520      30  09323N       2        SEQ
      P010601        UNC                190:ST1   071275 071275 071575 071575 18.334    520      26  09323N       2        SEQ
      BNET1          UNC                190:DP2   071275 071275 071275 071275 15.501    240     144  09323N       2        RAND
      NIF            UNC                190:DP2   070375 070375 080175 070375 11.612    250     241  09323N      12        SEQ
      M010101        UNC                190:DP2   071275 071275 071575 071575 18.335    520      26  09323N       2        SEQ
      T010202        UNC                190:DP2   071275 071275 071575 071575 18.335    520      26  09323N       3        SEQ
      T010101        UNC                190:DP2   071275 071275 071575 071575 18.335    520      29  09323N       2        SEQ
      T010203        UNC                190:DP2   071275 071275 071575 071575 18.335    520      26  09323N       2        SEQ
      T010602        UNC                190:DP2   071275 071275 071675 071675 09.582    520      26  09323N       4        SEQ
      E010401        UNC                190:DP2   071575 071575 071575 071575 13.177    260      13  09323N       4        SEQ
      E010501        UNC                190:ST1   071575 071575 071575 071575 13.270    180       9  09323N       1        SEQ
      M010201        UNC                190:ST1   071575 071575 071575 071575 13.271    300      15  09323N       1        SEQ
      M010301        UNC                190:ST1   071575 071575 071575 071575 13.311    120       6  09323N       1        SEQ
      M010302        UNC                190:ST1   071575 071575 071575 071575 13.311    380      19  09323N       1        SEQ
      M010303        UNC                190:ST1   071575 071575 071575 071575 13.312    320      16  09323N       1        SEQ
      M010304        UNC                190:DP2   071575 071575 071575 071575 14.102    200      10  09323N       2        SEQ
      M010305        UNC                190:DP2   071575 071575 071575 071575 14.102    160       8  09323N       2        SEQ
      P010301        UNC                190:DP2   071575 071575 071575 071575 17.021    340      17  09323N       9        SEQ
      M010306        UNC                190:DP2   071575 071575 071575 071575 15.450    220      11  09323N       2        SEQ
      M010307        UNC                190:DP2   071575 071575 071575 071575 15.450    140       7  09323N       1        SEQ
      M010402        UNC                190:DP2   071575 071575 071575 071575 15.450    280      14  09323N       1        SEQ
      M010403        UNC                190:DP2   071575 071575 071575 071575 15.501    240      12  09323N       2        SEQ
```

```
2574T 01  08-05-75    12.878            F I L E - S Y S T E M   A C T I V I T Y                        PAGE      2

                    2   CLIST AWCISS/CACI                                                          PRIOR STATUS ON PG

  CATALOG NAME   SCC         GEN'L DEVICE    CREATE       DATE OF LAST       TIME      MAX CURRENT LAST SAVE ALLOC  FILE
     FILE NAME               PERMS TYP:NAME   DATE   MODIFY ALLOC  CHANGE            LLINKS LLINKS VOL-CHG?  CNT  STAT TYPE

     M010404     UNC               190:DP2   071575 071575 071575 071575 15.501   280      14  09323N      2        SEQ
     M010501     UNC               190:ST1   071575 071575 071575 071575 17.772   400      20  09323N      3        SEQ
     M010502     UNC               190:ST1   071575 071575 071575 071575 17.855   220      11  09323N      1        SEQ
     M010601     UNC               190:DP2   071575 071575 071675 071675 10.176   260      13  09323N      4        SEQ
     M010602     UNC               190:ST1   071575 071575 071575 071575 17.855   180       9  09323N      2        SEQ
     P010101     UNC               190:DP2   071575 071575 071575 071575 17.918   460      23  09323N      2        SEQ
     P010102     UNC               190:DP2   071575 071575 071575 071575 17.947   480      24  09323N      1        SEQ
     P010201     UNC               190:DP2   071575 071575 071575 071575 17.947   260      13  09323N      1        SEQ
     P010302     UNC               190:DP2   071575 071575 071575 071575 17.998   380      19  09323N      1        SEQ
     P010303     UNC               190:DP2   071575 071575 071575 071575 17.998   220      11  09323N      1        SEQ
     P010401     UNC               190:DP2   071575 071575 071575 071575 18.037   740      37  09323N      1        SEQ
     P010701     UNC               190:DP2   071575 071575 071575 071575 18.037   420      21  09323N      1        SEQ
     P010801     UNC               190:DP2   071575 071575 071575 071575 18.118   160       8  09323N      1        SEQ
     P010901     UNC               190:DP2   071575 071575 071575 071575 18.082  1280      64  09323N      3        SEQ
     S010502     UNC               190:DP2   071575 071575 071575 071575 18.149   320      16  09323N      1        SEQ
     S010601     UNC               190:DP2   071575 071575 071575 071575 18.149   380      19  09323N      1        SEQ
     S010701     UNC               190:DP2   071575 071575 071575 071575 18.191   380      19  09323N      1        SEQ
     T010501     UNC               190:DP2   071575 071575 071575 071575 18.191   400      20  09323N      1        SEQ
     T010601     UNC               190:DP2   071575 071575 071575 071575 18.191   280      14  09323N      1        SEQ
     S010101     UNC               190:DP2   071575 071575 071575 071575 18.436   380      19  09323N      2        SEQ
```

TE

MED

82